PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2002-135243

(43)Date of publication of application : 10.05.2002

---

(51)Int.Cl. H04L 9/32
G06F 1/00
G06F 12/00
G06F 12/14
G06F 17/60
G09C 1/00
G10K 15/02
G10L 19/00
G10L 11/00
G11B 20/10
G11B 27/00
H04N 5/91
H04N 5/93

---

(21)Application number : 2000-320804 (71)Applicant : SONY CORP

(22)Date of filing : 20.10.2000 (72)Inventor : AKISHITA TORU
ISHIBASHI YOSHITO
YOSHINO KENJI
SHIRAI TAIZO

---

(54) DATA REPRODUCING APPARATUS, DATA RECORDING APPARATUS, DATA

REPRODUCING METHOD, DATA RECORDING METHOD, LIST UPDATING METHOD AND PROGRAM PROVIDING MEDIUM

(57)Abstract:

PROBLEM TO BE SOLVED: To provide a data processor which enables version management of a revocation list.

SOLUTION: Version information is set in a revocation list. For example when contents are read, the version of the revocation list which is held in a device is compared with a version of an effective revocation list existing a header of contents. When the version of the revocation list which is held is older, the reading of contents is stopped. When a revocation list for updating is compared with version information of a present revocation list. Only when it is judged that the list for updating is the newer revocation list, is updating of the revocation list allowed.

LEGAL STATUS [Date of request for examination]
[Date of sending the examiner's decision of rejection]
[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]
[Date of final disposal for application]
[Patent number]
[Date of registration]
[Number of appeal against examiner's decision of rejection]
[Date of requesting appeal against examiner's decision of rejection]
[Date of extinction of right]

CLAIMS

[Claim(s)]
[Claim 1] In the data regenerative apparatus which performs regeneration of the contents memorized by the data storage means The internal memory which stored the RIBOKESHON list which is a list of the data storage means or contents made applicable to processing prohibition which stored one of identifiers at least, and has the version information which shows old and new [of a list], The effective RIBOKESHON list version stored in the header information of the contents for playback, Comparison processing with the version of the RIBOKESHON list stored in said internal memory is performed. The version of the RIBOKESHON list stored in said internal memory The data regenerative apparatus characterized by having the controller which performs processing accompanying playback of said contents for playback a condition [a check not older than the version set as the header information of said contents for playback].
[Claim 2] The data storage means stored in the RIBOKESHON list stored in said internal memory as processing accompanying said playback or contents at least said controller One of identifiers, While having the configuration which performs comparison processing with the identifier of a data storage means which stored the contents which are for [of the contents which are the objects for playback / an identifier or for playback] The data storage means stored in said RIBOKESHON list in said comparison processing or contents at least One of identifiers, It is the data regenerative apparatus according to claim 1 characterized by being the configuration of performing processing which stops data playback when the identifier of the contents which are the objects for playback, or the identifier of a data storage means which stored the contents which are the objects for playback is in agreement.

[Claim 3] It is the data regenerative apparatus according to claim 1 which said controller has the memory interface which performs access to said data-storage means, and the control section which performs control of this memory interface, and is characterized by for said memory interface to be the configuration of performing comparison processing with the version of the effective RIBOKESHON list stored in the header information of the contents for playback, and the version of the RIBOKESHON list stored in said internal memory based on the data playback demand command from said control section.
[Claim 4] Said controller is the data regenerative apparatus according to claim 1 carry out having the configuration which performs comparison processing with the version of the RIBOKESHON list for updating received from the outside, and the version of the RIBOKESHON list stored in said internal memory, and performs an update process of the RIBOKESHON list by said RIBOKESHON list for updating a condition [it having been checked that the version of the RIBOKESHON list stored in said internal memory is newer than said RIBOKESHON list for updating] as the description.
[Claim 5] Said controller is a data regenerative apparatus according to claim 4 characterized by having the configuration which performs the data alteration check based on a data alteration check value (ICV) about the RIBOKESHON list for updating received from the outside, and performs an update process of the RIBOKESHON list by said RIBOKESHON list for updating based on the judgment without a data alteration.
[Claim 6] In the data recorder which performs record processing of the contents memorized for a data storage means The internal memory which stored the RIBOKESHON list which is a list of the data storage means or contents made applicable to processing prohibition which stored one of identifiers at least, and has the version information which shows old and new [of a list], As an effective RIBOKESHON list version stored in the header information of the contents for record The data recorder characterized by having the controller which performs processing which sets up the set point which directs the regeneration activation by refer to un-[RIBOKESHON list], and performs storing processing of the contents to said data storage means.
[Claim 7] Said controller is the data recorder according to claim 6 which has the memory interface which performs access to said data-storage means, and the control section which perform control of this memory interface, and is characterized by for said memory interface to be the configuration of perform the processing which sets up the version of the effective RIBOKESHON list which stores in the header

information of the contents for record as the set point by refer to un-[RIBOKESHON list] in which playback activation is possible based on the header-information generation command accompanying data logging from said control section.
[Claim 8] Said controller is the data recorder according to claim 6 carry out having the configuration which performs comparison processing with the version of the RIBOKESHON list for updating received from the outside, and the version of the RIBOKESHON list stored in said internal memory, and performs an update process of the RIBOKESHON list by said RIBOKESHON list for updating a condition [it having been checked that the version of the RIBOKESHON list stored in said internal memory is newer than said RIBOKESHON list for updating] as the description.
[Claim 9] Said controller is a data recorder according to claim 8 characterized by having the configuration which performs the data alteration check based on a data alteration check value (ICV) about the RIBOKESHON list for updating received from the outside, and performs an update process of the RIBOKESHON list by said RIBOKESHON list for updating based on the judgment without a data alteration.
[Claim 10] In the data playback approach in the data regenerative apparatus which performs regeneration of the data memorized by the data storage means The effective RIBOKESHON list version stored in the header information of the contents for playback, The comparison step which performs comparison processing with the version of the RIBOKESHON list stored in the internal memory of said data regenerative apparatus, The version of the RIBOKESHON list stored in said internal memory The data playback approach characterized by having the playback related processing execute step which performs processing accompanying playback of said contents for playback a condition [a check not older than the version set as the header information of said contents for playback].
[Claim 11] The data storage means stored in the RIBOKESHON list stored in said internal memory or contents at least said playback related processing execute step One of identifiers, The step which performs comparison processing with the identifier of a data storage means which stored the contents which are for [of the contents which are the objects for playback / an identifier or for playback], The data storage means stored in said RIBOKESHON list in said comparison processing or contents at least One of identifiers, It is the data playback approach according to claim 10 characterized by including the step which performs processing which stops data playback when the identifier of the contents which are the objects for playback, or the identifier of a data storage means which stored the contents which are the objects for playback is in agreement.

[Claim 12] The memory interface which performs access of as opposed to said data storage means in said data regenerative apparatus, It has the control section which performs control of this memory interface. Said data playback approach Furthermore, it sets to the step which transmits a data playback demand command from said control section to said memory interface, and said memory interface. The version of the effective RIBOKESHON list stored in the header information of the contents for playback based on reception of said data playback demand command, The data playback approach according to claim 10 characterized by including the step which performs comparison processing with the version of the RIBOKESHON list stored in said internal memory.
[Claim 13] The data-logging approach which carries out [having the step which performs the processing which sets up the set point which directs the regeneration activation by refer to un-/ RIBOKESHON list / in the data-logging approach of performing record processing of the contents memorized for a data-storage means, as an effective RIBOKESHON list version stored in the header information of the contents for record, and the step which perform storing processing of the contents to said data-storage means, and] as the description.
[Claim 14] It is the list of the data storage means or contents made applicable to processing prohibition which stored one of identifiers at least. The version of the RIBOKESHON list for updating which is the renewal approach of a list in the data processor which stored in the internal memory the RIBOKESHON list with the version information which shows old and new [of a list], and is received from the outside, Comparison processing with the version of the RIBOKESHON list stored in said internal memory is performed. The version of the RIBOKESHON list stored in said internal memory The renewal approach of a list characterized by performing an update process of the RIBOKESHON list by said RIBOKESHON list for updating a condition [it having been checked that it is newer than said RIBOKESHON list for updating].
[Claim 15] Said renewal approach of a list is the renewal approach of a list according to claim 14 characterized by having the step which performs the data alteration check based on a data alteration check value (ICV) further about the RIBOKESHON list for updating received from the outside, and performing an update process of the RIBOKESHON list by said RIBOKESHON list for updating based on the judgment without a data alteration.
[Claim 16] It is the program offer medium which offers the computer program which makes the data regeneration in the data regenerative apparatus which performs regeneration of the data memorized by the data storage means perform on computer

system. The effective RIBOKESHON list version by which said computer program was stored in the header information of the contents for playback, The comparison step which performs comparison processing with the version of the RIBOKESHON list stored in the internal memory of said data regenerative apparatus, The version of the RIBOKESHON list stored in said internal memory The program offer medium characterized by having the playback related processing execute step which performs processing accompanying playback of said contents for playback a condition [a check not older than the version set as the header information of said contents for playback].

DETAILED DESCRIPTION

[Detailed Description of the Invention]
[0001]
[Field of the Invention] This invention relates to a program offer medium at a data regenerative apparatus, a data recorder and the data playback approach, the data-logging approach, the renewal approach of a list, and a list. It is related with a program offer medium at the data regenerative apparatus which made possible version control of the RIBOKESHON list especially generated for the purpose of abatement of inaccurate media, inaccurate contents, etc., a data recorder and the data playback approach, the data-logging approach, the renewal approach of a list, and a list.
[0002]

[Description of the Prior Art] The rapid spread of the Internet in recent years and the negotiation which minded storages, such as networks of various software data (these are hereafter called contents (Content)), such as music data, a game program, and image data, such as the Internet, or DVD and CD, and a memory card, further with the spread of the small regenerator of a mobile mold, game machines, etc. are increasing rapidly. Contents regeneration or the program execution of these negotiation contents becomes possible by being stored in a storage or equipping the vessel only for playbacks, or a game device with storages, such as a memory card which was received from the network in PC (Personal Computer) which a user owns, the vessel only for playbacks, or the game device, and stored contents, and CD, DVD.
[0003] As a storage element of contents, a flash memory is in the component used recently. [many] A flash memory is one gestalt of nonvolatile memory rewritable on the electric target called EEPROM (Electrically Erasable Programmable ROM). Although the occupancy area per bit was large and the limitation was to make a degree of integration high since the conventional EEPROM constituted 1 bit from two transistors, the flash memory became possible [realizing 1 bit with one transistor with all bit package elimination methods]. The flash memory is expected as what can be replaced with record media, such as a magnetic disk and an optical disk.
[0004] The memory card which constituted the flash memory free [attachment and detachment] to data logging / playback device is also known. If this memory card is used, the digital audio record / regenerative apparatus which changes to disk-like media, such as the conventional CD (compact disk: trademark) and MD (mini disc: trademark), and uses a memory card are realizable.
[0005] When using such a contents storage element that used the flash memory in a personal computer (PC), a regenerator, etc., generally the file management system called a FAT (File AllocationTable) system is used as an access information table. If a required file is defined by the FAT system, a parameter required in it will be set in an order from the head of a file. Consequently, a file size can be made into variable length and one file can be constituted per 1 or two or more managements (a sector, cluster, etc.). The related matters of this management unit are written to the table called FAT. This FAT system can build a file structure easily regardless of the physical characteristic of a record medium. Therefore, a FAT system is employable also not only in a floppy (trademark) disk and a hard disk but a magneto-optic disk. The FAT system is adopted also in the memory card mentioned above.
[0006] By directions of the user through the input means connected [which were connected and was user-directed] from bodies of information machines and

equipment, such as a regenerative apparatus used as a playback device, a game device, and PC, various contents, such as music data, image data, or a program, are called from the flash memory mentioned above based on above-mentioned FAT, and are reproduced through the body of information machines and equipment or the connected display, a loudspeaker, etc.
[0007] Furthermore, generally as for many software contents, such as a game program, music data, and image data, the right of distribution etc. is held by the implementer and the vender. Therefore, it is common to permit the activity of software, and for reproduction without authorization etc. to be made not to be performed, namely, to take the configuration in consideration of security only to a fixed utilization limit, i.e., the user of normal, on the occasion of distribution of these contents.
[0008] One technique of realizing the utilization limit to a user is encryption processing of distribution contents. That is, while distributing various contents, such as voice data enciphered, for example through the Internet etc., image data, and a game program, it is a means, i.e., the configuration which gives a decode key, to decode the distributed encryption contents only to those who were checked as he is a registered user.
[0009] Encryption data can be returned to available decode data (plaintext) by decode processing in a predetermined procedure. The data encryption and the decryption approach of using an encryption key for encryption processing of such information, and using a decode key for decode processing are well learned from the former.
[0010]
[Problem(s) to be Solved by the Invention] In the contents record regenerator, the RIBOKESHON list is proposed as an approach for eliminating inaccurate media and inaccurate contents these days. The devices which perform record of contents and playback become possible [eliminating utilization of inaccurate contents] by performing processing which stops regeneration noting that they are inaccurate contents, when the identifier of the contents which stored contents for example, at the time of contents playback, and the contents identifier listed by the RIBOKESHON list are collated and an identifier in agreement is found.
[0011] However, processing of enabling playback of inaccurate contents may have been performed by altering a RIBOKESHON list or processing transposing the list set to the device to an unjust RIBOKESHON list etc. For example, a case so that the aggressor holding the invalidated inaccurate media or contents may hold all the time without updating the old RIBOKESHON list with which the inaccurate media or contents are not invalidated can be considered. If it does in this way, it will become

possible to perform the activity of inaccurate media and read-out of inaccurate contents which should be invalidated.
[0012] This invention is what offers the configuration which eliminates the alteration of such an unjust RIBOKESHON list, and updating. Specifically A version is set as a RIBOKESHON list. In the case of contents read-out The comparison with the version of the RIBOKESHON list currently held to the device and the version of the effective RIBOKESHON list in the header of contents is performed. By processing of enabling contents processing a condition [the version of a maintenance list not being old] It aims at providing with a program offer medium the data regenerative apparatus which made it possible to eliminate the unauthorized use of the contents by improper use of an unjust RIBOKESHON list, a data recorder and the data playback approach, the data-logging approach, the renewal approach of a list, and a list.
[0013]
[Means for Solving the Problem] In the data regenerative apparatus which performs regeneration of the contents the 1st side face of this invention was remembered to be by the data storage means The internal memory which stored the RIBOKESHON list which is a list of the data storage means or contents made applicable to processing prohibition which stored one of identifiers at least, and has the version information which shows old and new [of a list], The effective RIBOKESHON list version stored in the header information of the contents for playback, Comparison processing with the version of the RIBOKESHON list stored in said internal memory is performed. The version of the RIBOKESHON list stored in said internal memory It is in the data regenerative apparatus characterized by having the controller which performs processing accompanying playback of said contents for playback a condition [a check not older than the version set as the header information of said contents for playback].
[0014] In one embodiment of the data regenerative apparatus of this invention furthermore, said controller The data storage means stored in the RIBOKESHON list stored in said internal memory as processing accompanying said playback or contents at least One of identifiers, While having the configuration which performs comparison processing with the identifier of a data storage means which stored the contents which are for [of the contents which are the objects for playback / an identifier or for playback] The data storage means stored in said RIBOKESHON list in said comparison processing or contents at least One of identifiers, When the identifier of the contents which are the objects for playback, or the identifier of a data storage means which stored the contents which are the objects for playback is in agreement,

it is characterized by being the configuration of performing processing which stops data playback.
[0015] In one embodiment of the data regenerative apparatus of this invention furthermore, said controller It has the memory interface which performs access to said data storage means, and the control section which performs control of this memory interface. Said memory interface Based on the data playback demand command from said control section, it is characterized by being the configuration of performing comparison processing with the version of the effective RIBOKESHON list stored in the header information of the contents for playback, and the version of the RIBOKESHON list stored in said internal memory.
[0016] In one embodiment of the data regenerative apparatus of this invention furthermore, said controller Comparison processing with the version of the RIBOKESHON list for updating received from the outside and the version of the RIBOKESHON list stored in said internal memory is performed. It is characterized by the version of the RIBOKESHON list stored in said internal memory having the configuration which performs an update process of the RIBOKESHON list by said RIBOKESHON list for updating a condition [it having been checked that it is newer than said RIBOKESHON list for updating].
[0017] Furthermore, in one embodiment of the data regenerative apparatus of this invention, said controller is characterized by having the configuration which performs the data alteration check based on a data alteration check value (ICV) about the RIBOKESHON list for updating received from the outside, and performs an update process of the RIBOKESHON list by said RIBOKESHON list for updating based on the judgment without a data alteration.
[0018] Furthermore, the 2nd side face of this invention is set to the data recorder which performs record processing of the contents memorized for a data storage means. The internal memory which stored the RIBOKESHON list which is a list of the data storage means or contents made applicable to processing prohibition which stored one of identifiers at least, and has the version information which shows old and new [of a list], As an effective RIBOKESHON list version stored in the header information of the contents for record It is in the data recorder characterized by having the controller which performs processing which sets up the set point which directs the regeneration activation by refer to un-[RIBOKESHON list], and performs storing processing of the contents to said data storage means.
[0019] In one embodiment of the data recorder of this invention furthermore, said controller It has the memory interface which performs access to said data storage

means, and the control section which performs control of this memory interface. Said memory interface It is based on the header information generation command accompanying data logging from said control section. It is characterized by being the configuration of performing processing which sets up the version of the effective RIBOKESHON list stored in the header information of the contents for record as the set point by referring to un-[RIBOKESHON list] in which playback activation is possible.
[0020] In one embodiment of the data recorder of this invention furthermore, said controller Comparison processing with the version of the RIBOKESHON list for updating received from the outside and the version of the RIBOKESHON list stored in said internal memory is performed. It is characterized by the version of the RIBOKESHON list stored in said internal memory having the configuration which performs an update process of the RIBOKESHON list by said RIBOKESHON list for updating a condition [it having been checked that it is newer than said RIBOKESHON list for updating].
[0021] Furthermore, in one embodiment of the data recorder of this invention, said controller is characterized by having the configuration which performs the data alteration check based on a data alteration check value (ICV) about the RIBOKESHON list for updating received from the outside, and performs an update process of the RIBOKESHON list by said RIBOKESHON list for updating based on the judgment without a data alteration.
[0022] Furthermore, the 3rd side face of this invention is set to the data playback approach in the data regenerative apparatus which performs regeneration of the data memorized by the data storage means. The effective RIBOKESHON list version stored in the header information of the contents for playback, The comparison step which performs comparison processing with the version of the RIBOKESHON list stored in the internal memory of said data regenerative apparatus, The version of the RIBOKESHON list stored in said internal memory It is in the data playback approach characterized by having the playback related processing execute step which performs processing accompanying playback of said contents for playback a condition [a check not older than the version set as the header information of said contents for playback].
[0023] In one embodiment of the data playback approach of this invention furthermore, said playback related processing execute step The data storage means stored in the RIBOKESHON list stored in said internal memory or contents at least One of identifiers, The step which performs comparison processing with the identifier of a

data storage means which stored the contents which are for [of the contents which are the objects for playback / an identifier or for playback], The data storage means stored in said RIBOKESHON list in said comparison processing or contents at least One of identifiers, When the identifier of the contents which are the objects for playback, or the identifier of a data storage means which stored the contents which are the objects for playback is in agreement, it is characterized by including the step which performs processing which stops data playback.
[0024] In one embodiment of the data playback approach of this invention furthermore, said data regenerative apparatus It has the memory interface which performs access to said data storage means, and the control section which performs control of this memory interface. Said data playback approach Furthermore, it sets to the step which transmits a data playback demand command from said control section to said memory interface, and said memory interface. The version of the effective RIBOKESHON list stored in the header information of the contents for playback based on reception of said data playback demand command, It is characterized by including the step which performs comparison processing with the version of the RIBOKESHON list stored in said internal memory.
[0025] Furthermore, the 4th side face of this invention is to the data-logging approach which carries out as the description about having the step which performs the processing which sets up the set point which directs the regeneration activation by refer to un-[RIBOKESHON list] as an effective RIBOKESHON list version which stores in the header information of the contents for record, and the step which perform storing processing of the contents to said data-storage means in the data-logging approach of performing record processing of the contents which memorize for a data-storage means.
[0026] Furthermore, the 5th side face of this invention is the list of the data storage means or contents made applicable to processing prohibition which stored one of identifiers at least. The version of the RIBOKESHON list for updating which is the renewal approach of a list in the data processor which stored in the internal memory the RIBOKESHON list with the version information which shows old and new [of a list], and is received from the outside, Comparison processing with the version of the RIBOKESHON list stored in said internal memory is performed. The version of the RIBOKESHON list stored in said internal memory It is in the renewal approach of a list characterized by performing an update process of the RIBOKESHON list by said RIBOKESHON list for updating a condition [it having been checked that it is newer than said RIBOKESHON list for updating].

[0027] Furthermore, in one embodiment of the renewal approach of a list of this invention, it has the step which performs the data alteration check based on a data alteration check value (ICV) about the RIBOKESHON list for updating received from the outside, and is characterized by performing an update process of the RIBOKESHON list by said RIBOKESHON list for updating based on the judgment without a data alteration.
[0028] Furthermore, the 6th side face of this invention is a program offer medium which offers the computer program which makes the data regeneration in the data regenerative apparatus which performs regeneration of the data memorized by the data storage means perform on computer system. The effective RIBOKESHON list version by which said computer program was stored in the header information of the contents for playback, The comparison step which performs comparison processing with the version of the RIBOKESHON list stored in the internal memory of said data regenerative apparatus, The version of the RIBOKESHON list stored in said internal memory It is in the program offer medium characterized by having the playback related processing execute step which performs processing accompanying playback of said contents for playback a condition [a check not older than the version set as the header information of said contents for playback].
[0029] In addition, the program offer medium concerning the 6th side face of this invention is a medium which offers a computer program in a computer-readable format to the general purpose computer system which can perform various program codes, for example. Especially the gestalten, such as transmission media, such as record media, such as CD, and FD, MO, or a network, are not limited for a medium.
[0030] Such a program offer medium defines the collaboration-relation on the structure of the computer program and offer medium for realizing the function of a computer program predetermined in a computer system top, or a function. If it puts in another way, by installing a computer program in computer system through this offer medium, on computer system, a collaboration-operation is demonstrated and the same operation effectiveness as other side faces of this invention can be acquired.
[0031] The object, the description, and advantage of further others of this invention will become [rather than] clear by detailed explanation based on the example and the drawing to attach of this invention mentioned later.
[0032]
[Embodiment of the Invention] The contents distribution system configuration in which the application of the data processor of this invention to [system outline] drawing 1 is possible is shown. for example, contents, such as music data, image data,

and other various programs, store in the media 103 which are various record media, such as a memory card which carried CD, DVD, and the flash memory, through networks, such as the Internet, from a contents holder or a system management person 101 like a service provider -- having -- a device 102 -- reception -- or it is equipped, and reproduces and performs. A device has the display which is the device which has contents regenerative functions, such as a personal computer (PC), a vessel only for playbacks, and a game machine, for example, displays an image content, and the input unit which inputs directions of a user.

[0033] The detail configuration of the device which reproduces contents, and the media which store contents is shown in drawing 2 among such the contents distribution structure of a system.

[0034] Drawing 2 shows the detail configuration of a device 200, media 1,210, and media 2,230. Media 1,210 are media with the control section which supports only simple data read-out and write-in processing, and are media which have the controller which performs cipher processing of the contents which media 2,230 perform mutual recognition processing with the device equipped with media, and are stored in media. Wearing of as opposed to a device 200 in the both sides of media 1,210 and media 2,230 is possible.

[0035] The communications department 201 which performs data transceiver processing in which the device 200 of drawing 2 minded data communication means, such as the Internet, the input section 202 which inputs various directions, a message, The display 203 which performs the display of contents etc., and the control section 205 which performs these control, To a device controller 204 and a pan with the memory interface (I/F) section 300 with the interface function of data radial transfer with media, the file group of contents, It has the memory section 207 as an internal memory which stores the RIBOKESHON list as annulment information on inaccurate media or contents. In addition, data files, such as a RIBOKESHON list stored in an internal memory, are managed on a file quota table, and have the configuration which can be read.

[0036] It reproduces, after checking that the device 200 does not support the annulment media by which the contents for playback were stored in the RIBOKESHON list at the time of playback of contents, and annulment contents. When the contents for playback are listed by the RIBOKESHON list, it becomes a playback error and regeneration is not performed. The latter part explains the regeneration which applied the RIBOKESHON list and the RIBOKESHON list to a detail.

[0037] Media 1,210 have the control section 211 which controls data I/O, and the

memory section 212 which stores contents, and it the memory section 212 not only stores contents with response header information, but stores in media each BPT (Block Permission Table) which is Media ID and the access-permission table which described memory access control information further as identification information of a proper.
[0038] After the file system of a device 200 recognizes media, it reads BPT which is an access-permission table from media, and makes the memory interface section 300 which performs direct access to media transmit and manage BPT. The memory interface section 300 verifies an alteration check value (ICV) about BPT which received, after receiving BPT. Only when ICV is judged to be a just thing, BPT is saved as an effective thing. The memory interface section 300 performs only access based on BPT of these media, when the instruction which accesses the memory of media is received. The latter part explains the configuration of BPT, and the processing using BPT to a detail.
[0039] Media 2,230 are constituted by a controller 231 and the memory section 232, and the memory section 232 stores contents with response header information, and they store BPT (Block Permission Table) which is an access-permission table further. Data storage [as opposed to the memory section 232 in a controller 231] or the memory interface (I/F) section 234 as an interface for data read-out, media 2ID as an identifier of media, The preservation key Ksto which are the authentication key Kake applied to mutual recognition processing, and a cryptographic key at the time of the preservation to the memory section 232 of contents The internal memory 235 which stored initial value IV_keys when furthermore enciphering the key for encryption etc., authentication processing or encryption of contents, and decode processing are performed, and it has the cipher-processing section 236 equipped with the register, and the control section 233 which performs control of these each part.
[0040] The data storage configuration of each memory section of [the memory configuration in media] next media 1,210, and media 2,230 is shown in drawing 3 . The memory section is a flash memory which is called EEPROM (Electrically Erasable Programmable ROM) and which is one gestalt of rewritable nonvolatile memory electrically, and data elimination by the package elimination method of a block unit is performed.
[0041] As shown in drawing 3 (a), a flash memory has two or more blocks to the 1st - N, and as each block is shown in (b), it is constituted by two or more sectors to the 1-M, and each sector is constituted by the data division which contain live data as shown in (c), and the redundancy section containing redundancy data, such as an

error correction code. Although the latter part explains to a detail, ICV as a sector data alteration check value in the data division of each sector may be stored in the redundancy section.

[0042] [Main command] In the device 200 of drawing 2 , the main commands published in a control section 205 and the memory interface (I/F) section 300 are explained below.

[0043] First, there are the following in the command to the memory interface (I/F) section 300 from a control section 205.

- Read-out of the condition of a status register which set up the status within a status read-out command current memory interface. The memory interface (I/F) section 300 returns the content of the status register.
- The data read-out processing instruction of a sector which carried out sector read-out command assignment.
- The data write-in processing instruction to the sector which carried out sector write command assignment.
- The run command of the processing which decodes and reads the encryption data of the specified sector based on the information on the header by which the sector decode read-out commands set was carried out.
- The run command of the processing which enciphers and writes data in the specified sector based on the information on the header by which the sector code write-in commands set was carried out.
- The run command of the processing which generates a header based on the parameter by which header generation command assignment was carried out.
- The run command of the processing which sets a header set command header in a memory interface.
- The run command of the processing which sets the BPT set command BPT in a memory interface.
- RIBOKESHON list (Revocation List) set command inaccurate media, the run command of the processing which sets the RIBOKESHON list (Revocation List) which is a list of inaccurate contents in a memory interface.
- The run command of whether I may update a current RIBOKESHON list (Revocation List) on the RIBOKESHON list for renewal of a RIBOKESHON list (Revocation List) check command (Revocation List) for updating, and the processing to check.
- The run command of the processing to which the identifier (ID) of media is read to the media 1 by which media 1 recognition command connection was made, and the ID confirms whether to be validity or not.

– The run command of the processing to which mutual recognition is carried out to the media 2 by which media 2 recognition command connection was made, and the identifier (ID) of media confirms whether to be validity or not.
– The run command of the processing which reads the file quota table in file quota table call command memory.
– The run command of the processing which updates the file quota table to the renewal command memory of a file quota table.
[0044] The command to media 1 has the following from the memory interface (I/F) section 300.
– The run command of the processing which reads ID which ID read–out command media 1 have.
[0045] [Memory interface detail configuration in a device] The detail configuration of the memory interface (I/F) section 300 of a device 200 is shown in drawing 4 below. The function of each configuration section is explained.
[0046] – It is the register which saves the internal status of a status register 301 memory interface. The example of a configuration of a status register 301 is shown in drawing 5 . Each bit has following semantics.
– Bit 0 (bit 0) : busy flag (1: HIJI (busy) and 0:standby (ready)) It is the bit for distinction of whether the memory interface is carrying out internal processing.
– Bit 1 (bit 1) : read–out success flag (1: a success (success) and 0:failure (fail)) It is a bit for distinction with read–out of data successful from memory.
– Bit 2 (bit 2) : write–in success flag (1: a success (success) and 0: failure (fail)) It is a bit for distinction with the writing of data successful to memory.
– Bit 3 (bit 3) : one set flag of media (0: un–1: setting [a set (set) and] (not set)) It is a bit for distinction with the connected available media 1.
– Bit 4 (bit 4) : two set flag of media (0: un–1: setting [a set (set) and] (not set)) It is a bit for distinction with the connected available media 2.
– Bit 5 (bit 5) : media 1 effective flag (0: 1: validity (O.K.) and invalid (NG)) The identifier (ID) of the connected media 1 is that bit for distinction the outside for [in a RIBOKESHON list (Revocation List)] RIBOKU (abatement) media.
– Bit 6 (bit 6) : media 2 effective flag (0: 1: validity (O.K.) and invalid (NG)) The identifier (ID) of the connected media 2 is that bit for distinction the outside for [in a RIBOKESHON list (Revocation List)] RIBOKU (abatement) media.
– Bit 7 (bit 7) : Header set success flag (1: a success (success) and 0: failure (fail)) It is the bit for distinction of whether the header has set in the memory interface.
– Bit 8 (bit 8) : header generation success flag (1: a success (success) and 0: failure

(fail)) It is a bit for distinction with the successful generation of a header.
- Bit 9 (bit 9) : RIBOKESHON list (Revocation List) set flag (0: un-1: setting [a set (set) and] (not set)) It is the bit for distinction of whether the RIBOKESHON list (Revocation List) has set in the memory interface.
- Bit 10 (bit 10) : RIBOKESHON list (Revocation List) effective flag for updating (0: 1: validity (O.K.) and invalid (NG)) It is a bit for distinction with the effective RIBOKESHON list for updating (Revocation List).
[0047] A status register 301 holds the status information of these interface (I/F) sections 300.
[0048] Explanation is continued about return and the function of each configuration to drawing 4 .
- The register which sets up the total number of transfer sectors of register count register 304 data which set up the register address register 303 data-transfer initiation sector which saves the command transmitted from command register 302 control section [0049] In addition, the R/W of data to external memory and an internal memory sets the sector address which starts R/W as an address register, sets up the total number of sectors which write to a count register, and is performed by setting a sector R/W command to a command register.
[0050] - A register, transmit/receive control section 306 various registers, a transceiver buffer, etc. which set up actuation of a control register 305 memory interface perform control of a memory interface.
- The register for saving the data received for transmitting the data in the buffer and the transmit register 309 transmitting buffer memory 307 which stores the buffer and receiving buffer memory 308 received data which store transmitting buffer memory 307 transmit data a register and receive register 310, and transmitting to the receiving buffer memory 308 [0051] - Perform various cipher processing to the data in the cipher-processing section 320 transmitting buffer memory 307 and the receiving buffer memory 308.
- It is the field where the block permission table (BPT) as the RIBOKESHON list read from key information required for cipher processing in the memory section 321 cipher-processing section 320 and an internal memory and an access-permission table read from external memory is stored and saved. When a RIBOKESHON list and each block permission table (BPT) are set effectively in a memory interface and the transmit/receive control section 306 receives the media recognition command from a control section, or the R/W command of the data to external memory, set RIBOKESHON list and processing which referred to the block permission table (BPT)

are performed. The latter part explains these processings to a detail using a flow.
[0052] Furthermore, the following data are stored in the memory section 321 as key information required for cipher processing.
Kdist: The delivery key contained in the security header (Security Header) of contents other than the contents stored in media 2. Contents ICV generation key Kicv_cont and the contents key Kc are enciphered.
Kicv_sh : security header ICV generation key used in case ICV of a security header (Security Header) is generated.
IVsh: Initial value used in case ICV of a security header (Security Header) is generated (IV:Initial Value).
MKake: The master key for mutual recognition.
IVake: Initial value for applying to generation processing of the key for mutual recognition (IV:InitialValue).
IVauth: Initial value for the data generation at the time of mutual recognition (IV:Initial Value).
MKicv_rl: The master key which generates the ICV key of a RIBOKESHON list (Revocation List).
IVicv_rl : initial value when generating the ICV key of a RIBOKESHON list (Revocation List) (IV:Initial Value).
IVrl: Initial value used for the ICV generate time of a RIBOKESHON list (Revocation List) (IV:Initial Value).
IV_keys: Initial value when enciphering the key for contents encryption by media 2 (IV:Initial Value).
MKicv_bpt: The master key which generates the ICV key of BPT (Block Permission Table) which is access-permission information.
IVicv_bpt: Initial value used for the ICV generate time when generating the ICV key of BPT (Block Permission Table) which is access-permission information (IV:Initial Value).
IVbpt: Initial value of BPT (Block Permission Table) which is access-permission information (IV:Initial Value).
[0053] – It is the exclusive block which performs an ECC check about the data in the ECC circuit 323 transmit register 309 and a receive register 310.
[0054] – The input/output interface over external memory input/output interface 324 external memory (media 1 and 2). There is a memory card which carried the flash memory as external memory. For example, a block permission table (BPT) outputs and inputs through this external memory input/output interface to contents and the

header information accompanying contents record playback, and a pan.
- The input/output interface over internal-memory input/output interface 325 internal memory. I/O of for example, the RIBOKESHON list stored in the internal memory is performed through this interface.
[0055] From the external memory input/output interface 324 and the internal-memory input/output interface 325, each following signal is outputted to external memory (media 1 and 2) or an internal memory according to processing.
CLE: Enable [command latch enabling ALE:address latch / CE].: It is WP:write protect as a signal from chip enable WE:write enable RE:lead enabling and external memory (media 1 and 2), or an internal memory. (it applies only to external memory (media 1 and 2))
a RDY/BUSY:ready busy -- these various signals are inputted.
[0056] A [memory storing contents configuration], next the contents configuration stored in the flash memory of media are explained using drawing 6 . Each contents, such as music data and image data, are constituted by the security header which consists of various attribute information as shown in drawing 6 (a), and the contents as the live-data section.
[0057] The flash memory of media stores each pair of the security header unit of two or more contents, and the contents section, as shown in drawing 6 (b). As mentioned above, except for the case where make a flash memory into the gestalt which stores the security header unit or the contents section about the same contents in 1 block since elimination is performed per block, and the elimination processing put in block is permitted, processing which stores different contents in one block is not performed.
[0058] A [security header configuration] security header is the attribute information corresponding to each contents. The data configuration of a security header is shown in drawing 7 . Each content of data is explained.
[0059] - Format version (Format Version)
The format version of a security header (Security Header) is shown.
- Content ID (Content ID)
The identifier (ID) of contents is shown.
- Contents type (Content Type)
The class of contents is shown. For example, they are the contents stored in media 1 or media 2, or broadcast contents.
- Data type (Data Type)
It is shown whether they are data, such as the attribute of contents, for example, music, and an image, or it is a program.

- Cryptographic algorithm (Encryption Algorithm)
The encryption processing algorithm using the contents key (Kc) of contents is shown. For example, it is shown whether it is encryption by DES or it is based on Triple DES (Triple-DES).
- Encryption Mohd (Encryption Mode)
Code Mohd corresponding to the algorithm specified by encryption algorithm (Encryption Algorithm) is shown. For example, ECB Mohd, CBC Mohd, etc. is shown.
[0060] - Encryption format type (Encryption Format Type)
An encryption format of contents is shown. Type 1, Type 2, and the type enciphered with one contents key Kc to the whole contents is considered as Type 1, and the mode which enciphers contents with the application of different key Ksec_n for every sector of contents is considered as Type 2.
[0061] The encryption format configuration of each type is shown in drawing 8 . Drawing 8 (a) is the memory storing configuration of the contents enciphered in the encryption format of Type 1, and (b) is the memory storing configuration of the contents enciphered in the encryption format of Type 2.
[0062] An encryption format of Type 1 shown in drawing 8 (a) is the configuration which contents were altogether enciphered using one contents key Kc, and was stored in memory, i.e., sector non-dependence encryption processing. An encryption format of Type 2 shown in drawing 8 (b) is the configuration in which the contents as which different sector key Ksec_1 for every sector of a flash memory - Ksec_m were applied and enciphered were stored, i.e., sector-dependent encryption processing. For example, with the sector 1 of the flash memory of drawing 8 (b), in each block, encryption processing which applied Ksec_1 altogether is performed and the contents by which Ksec_1 corresponds, it is set up and stored in a sector 1 as an encryption key of a sector 1 are stored. With the sector m of a flash memory, in each block, encryption processing which applied Ksec_m altogether is performed and the contents by which Ksec_m corresponds, is set up and stored in Sector m as an encryption key of Sector m are stored.
[0063] Thus, in the configuration of this invention, cipher processing of the contents which applied a different encryption key for every sector is applied. Furthermore, also in the processing mode which applied a different encryption key for every sector, various kinds of encryption modes, such as processing by the single DES which applied one key to one sector, and processing by the Triple DES which applied two or more keys to one sector, are applicable. The latter part explains these processings to a detail further.

[0064] Explanation is continued about the configuration of return and a security header to drawing 7 .
- Encryption flag (Encryption Flag)
The flag which shows encryption and un-enciphering. [of each sector within a block] It has a flag for the number of sectors (for example, 32 sectors) within a block. For example, 0:non-enciphering sector, 1: Encryption sector. In addition, let 1 block be 32 sectors in this example.
[0065] - ICV flag (ICV Flag)
The flag which shows ICV addition and un-adding. [of each sector within a block] It has a flag for the number of sectors (32 sectors) within a block. For example, 0:ICV nothing, those with 1:ICV [0066] - Encryption contents key (Kc_Encrypted 0-31)
The storing field (32 pieces) and encryption ICV generation key of the enciphered contents key (Kicv_cont_encrypted)
The storing field of the key for ICV creation of the enciphered contents [0067] - Effective RIBOKESHON list version (Valid Revocation List version)
The version of the RIBOKESHON list (Revocation List) applied effectively in the case of contents playback. Playback is not permitted when the version of the RIBOKESHON list (RevocationList) set in the case of contents playback is older than this. In addition, regeneration of the data stored in the self-device etc. sets 0 to the contents which do not need to apply reference of a RIBOKESHON list.
[0068] - Security header ICV (ICV of Security Header)
The alteration check value of a security header (Security Header) (ICV).
[0069] The configuration of a [RIBOKESHON list], next the RIBOKESHON list as annulment information on inaccurate media or contents is explained. The configuration of a RIBOKESHON list is shown in drawing 9 . Hereafter, each data is explained.
[0070] - RIBOKESHON list identifier (Revocation List ID)
It is ID as an identifier of a RIBOKESHON list (Revocation List) proper.
[0071] - RIBOKESHON list version (Revocation List Version)
The version of a RIBOKESHON list (Revocation List) is shown. A RIBOKESHON list is updated and adds the annulment information on new inaccurate media or contents at the time of updating.
[0072] With the configuration of this invention, version information is set as a RIBOKESHON list (Revocation List), and the version information of the effective RIBOKESHON list in the header of contents is set up. In the case of contents read-out, the version of the RIBOKESHON list currently held to the current device is

compared with the version of the effective RIBOKESHON list in the header of contents. Under the present circumstances, when the version of the RIBOKESHON list which is carrying out current maintenance is older, read-out of contents is stopped. Consequently, if a RIBOKESHON list is not updated, read-out of the contents cannot be performed.
[0073] Moreover, the memory interface section compares the version information of a current RIBOKESHON list with the version information of the RIBOKESHON list for updating at the time of renewal of a RIBOKESHON list, and only when it judges that it is a new RIBOKESHON list, it considers as the configuration to which renewal of a RIBOKESHON list is permitted.
[0074] The latter part explains the example of concrete processing of old and new comparison processing of the RIBOKESHON list using version information, and an update process to a detail using a processing flow.
[0075] - The number of media 1ID (Number of Media1 ID)
Total and media 1ID(0)-media 1ID (L-1) (Media1ID(0)-Media1ID (L-1)) of the invalidated media 1 (Media1 ID)
It is a list of identifiers of the invalidated media 1.
[0076] - The number of media 2ID (Number of Media2 ID)
Total and media 2ID(0)-media 2ID of the invalidated media 2 (Media2 ID) (M-1) (Media2ID(0)-Media2ID (M-1))
It is a list of identifiers of the invalidated media 2.
[0077] - The number of content ID (Number of Contents ID)
The total and content ID (0)-content ID of the invalidated content ID (Contents ID) (N-1) (Contents ID(0)-Contents ID (N-1))
It is a list of invalidated contents identifiers.
[0078] - RIBOKESHON list ICV (ICV of Revocation List)
ICV for the alteration check of a RIBOKESHON list [0079] As mentioned above, the RIBOKESHON list in this invention consists of identifiers (ID) of two or more classes (media, contents). Thus, it becomes possible to eliminate two or more contents by one RIBOKESHON list, and media by preparing two or more media ID for [ID] RIBOKU and content ID of a class in the RIBOKESHON list (Revocation List) which is the annulment information on contents or media, and performing each collating as different actuation. The activity of inaccurate media and read-out of inaccurate contents can be forbidden by performing collating with ID listed by the RIBOKESHON list with the identifier (ID) of utilization media or utilization contents in the memory interface section at the time of insertion of media, and read-out of contents.

[0080] Thus, the media of two or more classes by one RIBOKESHON list and RIBOKU (abatement) of contents become possible by the configuration of having set two or more ID of contents or media as one RIBOKESHON list. The latter part explains the verification processing of media based on the RIBOKESHON list of [at the times of media starting], and concrete processing of the contents verification processing at the time of contents processing.
[0081] Moreover, with the configuration of this invention, since the RIBOKESHON list was set up in the memory interface which carries out direct access to external memory etc. and after the setup considered it as the continuously available configuration in the memory interface in the time of playback of contents at the time of wearing of media, processing of reading from an internal memory to the utilization time of contents repeatedly becomes unnecessary, and processing is performed efficiently.
[0082] The configuration of [a block permission table (BPT)], next the block permission table (BPT:Block Permission Table) used as an access-permission table is explained. When playback of contents was performed in the former, for example, PC etc., the file system of OS in PC had read and managed actively the access information table (for example, File Allocation Table; FAT) stored in the archive medium, and, as for the file system, rewriting was able to do the content of the access information table freely. Therefore, even if there is an archive medium which stores the access information table which set up the write inhibit, when a file system rewrites the access information table as reading, the data in an archive medium may be rewritten.
[0083] The block permission table (BPT) adopted in the data processor of this invention is own access-permission table of media stored in the block which forbade rewriting in a device. When data processing, such as contents data writing, is performed using the media which stored BPT, a device is setting a block permission table (BPT) to the memory interface section of the device which carries out direct access to media, and was taken as the configuration to which memory access according to the authorization information set as the block permission table (BPT) whose control section of a device is the access-permission table of media also in activation about what kind of program is carried out.
[0084] The configuration of a block permission table (BPT) is shown in drawing 10 . Hereafter, each data is explained.
[0085] - Format version (Format Version)
The format version of BPT (Block Permission Table) is shown. It is data which the

BPT itself has various kinds of formats, and identify the any they are.
- BPT identifier (BPT ID)
It is the identifier (ID) of a block permission table (BPT:Block Permission Table).
- Block count (Number of Blocks)
The total of the block treated by BPT (Block Permission Table) is shown. As mentioned above, as for a flash memory, elimination for every block is made. The block count managed by BPT is shown.
- Block #1-block #n authorization flag (Block #1-#n Permission Flag)
The access-restriction flag of each block is shown. For example, it is shown that the block of a flag 0 is an elimination improper block, and the block of a flag 1 is an elimination good block.
- BPT-ICV (ICV of BPT)
It is ICV for the alteration check of BPT (Block Permission Table).
[0086] After the file system of a device recognizes a device, it reads a block permission table (BPT) from media, such as a memory card which carried the flash memory, transmits BPT to the memory interface section which performs direct access to media, and is made to manage it as an access-permission table to the media. The memory interface section receives an access-permission table, and sets BPT (memory section 321 shown in ex. drawing 4). A memory interface performs only access based on the access-permission table of these media, when the instruction which accesses the memory of media is received.
[0087] The permitted processing mode in each block unit of the flash memory of Media and setting out which is specifically an elimination good block, an elimination improper block or a playback good block, a playback improper block, etc. are made by the block permission table (BPT). A memory interface determines the propriety of processing according to these BPT setting out. The latter part explains the detail of these processings to a detail further.
[0088] In addition, the alteration check value ICV for alteration prevention is set up, and at the time of the set to the memory interface of BPT, when an ICV check is performed by the block permission table (BPT) and it is judged with those with an alteration, set processing of BPT is not performed on it. Therefore, creating and using an inaccurate access-permission table is prevented. ICV of BPT is generated based on the identifier (ID) of media. Therefore, the media cannot be used even if it copies an access-permission table to other media. About generation of ICV, it mentions later.
[0089] Media write in and ship a block permission table (BPT) to the predetermined block of memory (ex. flash memory) at the time of the manufacture. Under the present

circumstances, about the block in the memory which stored the block permission table (BPT), setting out for which block elimination is improper is described on a block permission table (BPT). Since the device of this invention is the configuration of performing elimination of only an eliminable block after referring to the elimination propriety of each block set as BPT with reference to BPT in the data elimination processing stored in media, elimination of BPT and a rewrite substitute are prevented about the media set up as elimination of a BPT storing block being impossible. About the writing of the file using BPT in media, and regeneration, it mentions later.
[0090] The setting-out flow of the block permission table (BPT) at the time of manufacture of media (flash memory loading data-logging medium) is shown in drawing 11 and drawing 12 . Here, generation of a media identifier (ID) and the writing of BPT shall be performed by continuous action through media and the media creation machine which can perform a command communication link.
[0091] Drawing 11 is the setting-out flow of the block permission table (BPT) which the media creation machine in the type without a mutual recognition processing facility of media 1 performs. Each processing is explained. first, to the media to which initial setting is not performed yet, ID reading appearance is carried out, a command is sent (S31), and if ID beforehand stored in media is received (S32), ICV generation key Kicv_bpt which used the ID as the base will be generated (S33). ICV generation key Kicv_bpt is generated based on master key:MKicv_bpt, initial value:IVicv_bpt, and a BPT identifier (ID). Specifically based on ICV generation key Kicv_bpt=DES (E, MKicv_bpt, ID^IVicv_bpt), it is generated. The semantics of a formula means that encryption processing by DES Mohd by master key:MKicv_bpt is performed to ID of BPT, and the exclusive OR of initial value IVicv_bpt.
[0092] Next, ICV which generated and (the configuration of drawing 14 mentioned later is applied) (S35) generated ICV based on BPT to which the parameter required for each field of BPT was set (S34), and each parameter was set is set as the ICV field of BPT (S36). Thus, the constituted block permission table (BPT) is written in media 1 (S37). In addition, as mentioned above, the write-in block of BPT is considered as the block set up as an elimination improper field in BPT.
[0093] Drawing 12 is the setting-out flow of the block permission table (BPT) which the media creation machine in the type with a mutual recognition processing facility of media 2 performs. Each processing is explained. First, mutual recognition processing with the media 2 to which initial setting is not performed yet, and sharing (see the processing of drawing 22 mentioned later about these processings) of a session key are performed.

[0094] After mutual recognition and key share processing are completed, ID read-out command is sent to media 2 (S41), ID is read, and ICV generation key Kicv_bpt which used ID as the base is generated (S42). ICV generation key Kicv_bpt is generated based on master key:MKicv_bpt, initial value:IVicv_bpt, and a BPT identifier (ID). Specifically based on ICV generation key Kicv_bpt=DES (E, MKicv_bpt, ID^IVicv_bpt), it is generated. The semantics of a formula means that encryption processing by DES Mohd by master key:MKicv_bpt is performed to ID of BPT, and the exclusive OR of initial value (IVicv_bpt).

[0095] Next, ICV which generated and (the configuration of drawing 14 mentioned later is applied) (S46) generated ICV based on BPT to which the parameter required for each field of BPT was set (S45), and each parameter was set is set as the ICV field of BPT (S47). Thus, the constituted block permission table (BPT) is written in media 1 (S48). In addition, as mentioned above, the write-in block of BPT is considered as the block set up as an elimination improper field in BPT.

[0096] The example of a concrete configuration of a block permission table (BPT) is shown in drawing 13 . (a) of drawing 13 is the block configuration of the flash memory of media 1 and media 2, and drawing 13 (b) is a block permission table (BPT). a block permission table (BPT) -- a format version, BPTID, and the block count -- then, elimination of each block is possible -- (1) and elimination are impossible -- (0) is set up and it has the configuration in which the alteration check value (ICV of BPT) of BPT was finally stored. The BPT storing block (the example of drawing 13 block # 2) of memory is set up as an elimination improper field in a block permission table (BPT), prevents elimination by the device, and has the configuration with which rewriting of BPT is not performed.

[0097] in addition, the example of a configuration of the block permission table (BPT) shown in drawing 13 is [of each block] eliminable -- (1) and elimination are impossible -- although it is the configuration that (0) was set up, it is good also as not the configuration that sets up the access permission of only elimination processing but a configuration which read (playback) and set up authorization and disapproval. for example, playback and elimination are impossible -- (11), a playback good, and elimination are impossible -- (10), playback improper, and elimination are possible -- (01), playback, and elimination are possible -- setting out set to (00) is possible.

[0098] In addition, even if the new write-in instruction of BPT comes by the condition that have a control section 231 in media by media 2, and can also hold the condition of being finishing [a block permission table (BPT) / setting out] as shown in drawing 2 , and BPT is set up, from a device, it is good as a configuration which is not received

also as a configuration which prevents the re-writing of BPT.
[0099] In addition, although the BPT writing in an above-mentioned example explained the configuration performed through media and the media creation machine which can perform a command communication link, its writing of BPT to media is good also as a configuration which writes in directly BPT created with the simple memory writer. However, the BPT storing block of memory is set up as an elimination improper field in a block permission table (BPT) also in this case.
[0100] [The alteration check by the alteration check value (ICV)], next the data alteration check processing by the alteration check value (ICV:Integrity Check Value) are explained. In the configuration of this invention, an alteration check value (ICV) is added to the contents stored in a data storage means, a block permission table, a RIBOKESHON list, etc., and is applied to each data alteration check processing. In addition, the alteration check value about contents is the configuration which can be added to a sector data unit. The latter part explains the concrete gestalt of the ICV processing added to contents, the block permission table, the RIBOKESHON list, etc.
[0101] The example of alteration check value (ICV) generation using a DES cipher-processing configuration is shown in drawing 14 . The message which constitutes the target alteration check data as shown in the configuration of drawing 14 is divided per 8 bytes (the divided message is hereafter set to D0, D1, D2, ..., Dn-1). Alteration check data are configuration data of BPT which is the access-permission table which was for example, the contents itself or was mentioned above, or are configuration data of a RIBOKESHON list.
[0102] First, the exclusive OR of D0 is carried out to initial value (Initial Value (hereafter referred to as IV)) (the result is set to I1). Next, I1 is put into the DES encryption section, and it enciphers using the alteration check value (ICV) generation key Kicv (an output is set to E1). Continuously, the exclusive OR of E1 and D1 is carried out, the output I2 is put in to the DES encryption section, and it enciphers using the alteration check value (ICV) generation key Kicv (output E2). Hereafter, this is repeated and encryption processing is performed to all messages. EN which came out at the end is made into contents check value ICV'.
[0103] If it is guaranteed that that there is no alteration compares guaranteed just ICV which was generated, for example to the contents generate time with ICV' newly generated based on contents, and there will be no alteration in an incoming message, for example, contents, BPT, or a RIBOKESHON list if identity is demonstration, i.e., ICV'=ICV, and it is ICV'!=ICV, it will be judged with there having been an alteration.
[0104] The data alteration check processing flow which used ICV is shown in drawing

15 . First, the object data of an alteration check are extracted (S11), and ICV' is calculated by the DES cipher-processing configuration of drawing 14 based on the extracted data (S12). Computed ICV' is compared with ICV stored in data as a result of count (S13), and when in agreement, there is no alteration of data, it is judged in case of just data (from S14 to S15), and in the case of an inequality, it will be judged if there is an alteration of data (from S14 to S16).

[0105] Alteration check value (ICV) generation key Kicv_rl for the alteration check of a RIBOKESHON list Master key:MKicv_rl which generates the ICV key of the RIBOKESHON list (Revocation List) beforehand stored in the memory section 321 (refer to drawing 4) of the memory interface section 300 of a device, Initial value when generating the ICV key of a RIBOKESHON list (Revocation List): Generate based on the RIBOKESHON list version (Version) contained in IVicv_rl and the attribute information on a RIBOKESHON list. Specifically based on alteration check value (ICV) generation key Kicv_rl=DES (E, MKicv_rl, Version^IVicv_rl), it is generated. The semantics of said formula means that encryption processing by DES Mohd by master key:MKicv_rl is performed to the exclusive OR of a version (Version) and initial value (IVicv_rl). The alteration check value of a RIBOKESHON list is performed by the ICV generation configuration shown in drawing 15 using initial value IVrl (it stores in the memory section 321) with the application of ICV generation key Kicv_rl generated by doing in this way.

[0106] Moreover, alteration check value (ICV) generation key Kicv_bpt for the alteration check of a block permission table (BPT) is generated based on the BPT identifier (ID) contained in master key:MKicv_bpt which generates the ICV key of BPT beforehand stored in the memory section 321 (refer to drawing 4) of the memory interface section 300 of a device, initial value:IVicv_bpt when generating the ICV key of BPT, and the attribute information on BPT. Specifically based on alteration check value (ICV) generation key Kicv_bpt=DES (E, MKicv_bpt, ID^IVicv_bpt), it is generated. The semantics of said formula means that encryption processing by DES Mohd by master key:MKicv_bpt is performed to ID of BPT, and the exclusive OR of initial value (IVicv_bpt). The alteration check value of a block permission table (BPT) is performed by the ICV generation configuration shown in drawing 15 using initial value IVbpt (it stores in the memory section 321) with the application of ICV generation key Kicv_bpt generated by doing in this way. In addition, ICV stored as incidental information on BPT is generated based on the data in BPT, and the data containing the identifier (ID) of the media which stored BPT. Therefore, the ICV check of BPT has not only the existence of a data alteration of BPT but the function to verify that it is not BPT with

a just media proper, i.e., BPT copied to other media.
[0107] moreover , alteration check value (ICV) generation key Kicv_cont for the alteration check of the sector unit of contents be encipher and store in the header (security header) of contents , and be acquire by the decode processing by DES-CBC Mohd perform by the controller 231 of the media 2 perform after mutual recognition with media 2 in the cipher processing section 320 (refer to drawing 4) of a memory interface if needed . It is during the explanation which used the flow about these processings, and explains to a detail.
[0108] Processing which will forbid processing of the playback of contents based on reference processing of a RIBOKESHON list etc. if it becomes clear as a result of such a data alteration check (for example, the alteration of a RIBOKESHON list), and will forbid access to the data of the media based on BPT if judged with BPT which is an access-permission table having an alteration is performed. The latter part explains these processings to a detail.
[0109] In the data processor of this invention, the processing performed when processing in case a device performs data read-out from media, and a device store data to media is explained below [data read-out and write-in processing].
[0110] (It processes at the time of device starting) The processing at the time of starting a device is first explained using drawing 16 . Drawing 16 shows processing of the memory interface section 300 to left-hand side on processing of the control section 205 of the device 200 in drawing 2 , and right-hand side. The conditions of the status register of the memory interface section 300 in a processing start event are busy flag:0 (standby) and RIBOKESHON list set flag:0 (un-setting).
[0111] First, if a device starts, a control section will transmit the file quota table call command of an internal memory to the memory interface section (S101). The memory interface section transmits the read-out command of a file quota table to the internal memory of a device (S102), receives a file quota table from an internal memory, and transmits to a control section (S103).
[0112] In addition, a file quota table has the configuration with which the directory, the file name, and the storing sector were matched, as it is the table which carries out directory management of the various data files, such as the data stored in the accessible internal memory of a device, and external memory, for example, various contents, and a RIBOKESHON list, for example, is shown in drawing 17 . A device accesses various files based on a file quota table.
[0113] If the file quota table corresponding to the data stored in the internal memory is received (S104), a control section will perform read-out processing of a

RIBOKESHON list based on a table (S105), and will transmit a RIBOKESHON list to a memory interface with the set command of a RIBOKESHON list (S106). Set processing of a RIBOKESHON list will perform comparison processing with the contents or the media identifier listed by the RIBOKESHON list in the case of contents processings, such as contents read-out processing from media, if it performs only when a RIBOKESHON list is effective, and a list is set. About these processings, it mentions later.
[0114] If a RIBOKESHON list is received from a control section with the set command of a RIBOKESHON list (S107), a memory interface will set the busy flag of a status register to 1 (busy) (S108), and will generate alteration check value (ICV) generation key Kicv_rl for the alteration check of a RIBOKESHON list (S109).
[0115] Alteration check value (ICV) generation key Kicv_rl for the alteration check of a RIBOKESHON list is generated based on the RIBOKESHON list version (Version) contained in master key:MKicv_rl which generates the ICV key of the RIBOKESHON list (Revocation List) beforehand stored in the device, initial value:IVicv_rl when generating the ICV key of a RIBOKESHON list (Revocation List), and the attribute information on a RIBOKESHON list. Specifically based on alteration check value (ICV) generation key Kicv_rl=DES (E, MKicv_rl, Version^IVicv_rl), it is generated. The semantics of a formula means that encryption processing by DES Mohd by master key:MKicv_rl is performed to the exclusive OR of a version (Version) and initial value (IVicv_rl).
[0116] Next, a memory interface generates ICV' of a RIBOKESHON list using generated alteration check value (ICV) generation key Kicv_rl, and performs collating processing (ICV'=ICV?) with the right ICV beforehand stored in the RIBOKESHON list (S110). In addition, generation processing of ICV' is performed by the processing which applied generated alteration check value (ICV) generation key Kicv_rl using initial value IVrl based on DES Mohd who explained by above-mentioned drawing 14 .
[0117] When it is ICV'=ICV (it is Yes at S111), it is judged with it being a just thing without an alteration of a RIBOKESHON list, and sets to the condition which can be referred to in the cases, such as read-out processing of contents, and a RIBOKESHON list set flag is set to 1 (set) (S112). A RIBOKESHON list is stored in the memory within a memory interface (for example, memory section 321 (refer to drawing 4)). For example, the media identifier of the RIBOKESHON list set when the transmit/receive control section 306 received the media recognition command from the control section 205 (refer to drawing 2), Collating with the media identifier of the media with which the device was equipped is performed. Moreover, collating with the

contents identifier of the RIBOKESHON list set when the transmit/receive control section 306 received the header set command accompanying read-out processing of contents from the control section 205, and the contents identifier of the contents for read-out is performed.
[0118] Thus, a RIBOKESHON list is set up in the memory interface which carries out direct access to external memory etc., after a setup, at the time of wearing of media, in the time of playback of contents, it considers as a continuously available configuration in a memory interface, processing of reading from an internal memory to the utilization time of contents repeatedly becomes unnecessary, and processing is performed efficiently.
[0119] Explanation of the flow of drawing 16 is continued. When it is ICV'!=ICV (it is No at S111), it is judged with those with an alteration by the RIBOKESHON list, the contents processing based on reference processing of a list is forbidden, and processing is ended. A busy flag is set to 0 by termination of the above processing.
[0120] On the other hand, a control-section side transmits a status read-out command to a memory interface (S114), and saves a RIBOKESHON list set flag by making to have set the busy flag to 0 into conditions (S115) (S116). In the case of 1 which shows that the list was set effectively, and others, it is set to 0 when judged with the RIBOKESHON set flag saved not having the alteration of a list.
[0121] (It processes at the time of media recognition) Next, the processings performed at the time of media recognition, such as an effectiveness check of media when a device is equipped with media, are explained. As mentioned above, there are the media 1 of a type which do not perform mutual recognition processing with a device, and the media 2 of a type which perform mutual recognition processing with a device as media. A device [whether if a device is equipped with each type, contents processing using media may be performed, and] Processing which checks whether there is specifically any registration as inaccurate media in a RIBOKESHON list is performed. Wearing media are not listed by the RIBOKESHON list but it is contingent [on it having been checked that they are available media effectively]. BPT (Block Permission Table) which is the access-permission table stored in media is set to a memory interface, and processing which makes possible memory access which referred to BPT is performed.
[0122] First, the media check processing at the time of being equipped with media 1 is explained using drawing 18 and drawing 19 .
[0123] Also in drawing 18 and drawing 19 , processing of the memory interface section 300 is shown in left-hand side on processing of the control section 205 of the device

200 in drawing 2 , and right-hand side. this flow initiation event -- it is -- the condition of the status register of the memory interface section 300 -- busy flag:0 (standby) and media 1 -- effective -- it is in the condition of flag:0 (invalid) and media 1 set flag:0 (un-setting).
[0124] First, a control section recognizes that the media with which the device was equipped are media 1 (S201). Media discernment is performed based on the communication link information between the mechanical information based on the media configuration set up beforehand or a device, and media. If it recognizes that control sections are media 1, a control section will transmit a media 1 recognition command to a memory interface (S202).
[0125] If the media 1 recognition command from a control section is received (S203), a memory interface sets the busy flag of a status register as 1 (busy) (S204), to media 1, will transmit the read-out command of the identifier (ID) of media 1 (S205), and will receive it (S206). Furthermore, comparison collating with ID of the media 1 which received, and the list of the RIBOKU (abatement) media 1 under already set RIBOKESHON list is performed (S207). As the RIBOKESHON list was explained in the flow at the time of starting of previous drawing 16 , it is set up in a memory interface at the time of starting, and after a setup becomes continuously available in a memory interface in the time of playback of contents at the time of wearing of media.
[0126] When it does not exist while ID which is in agreement with Reception ID listed, if the wearing media 1 are not the media for RIBOKU but media available to validity, they will be judged (it sets to S208 and is No), they set the media 1 effective flag of a status register to 1 (effective) (S209), and set a busy flag to 0 (standby) (S210). When ID which is in agreement with Reception ID is during a RIBOKESHON list (it sets to S208 and is Yes), the wearing media 1 are the media for RIBOKU, they judge with the ability not to use effectively, and do not perform validation processing of the effective flag of step S209, but set a busy flag to 0 (standby) at step S210, and end processing.
[0127] On the other hand, in step S211, a control section transmits a status read-out command to a memory interface, and a media flag condition is checked, only when it is validity (flag: 1) (it is Yes at S213), processing is continued, and after checking that the busy flag has been set to 0 (standby) (S212), when it is an invalid (flag: 0) (it is No at S213), it ends processing.
[0128] Next, it progresses to drawing 19 and a control section transmits the file quota table call command about media 1 to a memory interface (S221), and a memory interface transmits the sector read-out command with which the file quota table was stored to media 1 (S222), it receives a file quota table from media 1, and transmits it

to a control section (S223).
[0129] If the file quota table corresponding to the data stored in media 1 is received (S224), a control section will perform read-out processing of a block permission table (BPT) based on a table (S225), and will transmit the set command of BPT, and BPT to a memory interface (S226). Set processing of BPT will judge whether elimination for every block is possible with reference to BPT in the case of contents processings, such as contents write-in processing from media, if it performs only when BPT is effective, and BPT is set. The latter part explains data write-in processing in which actual BPT was referred to.
[0130] If the set command of a block permission table (BPT) and BPT are received from a control section (S227), a memory interface will set the busy flag of a status register to 1 (busy) (S228), and will generate alteration check value (ICV) generation key Kicv_bpt for the alteration check of BPT (S229).
[0131] Alteration check value (ICV) generation key Kicv_bpt for the alteration check of BPT is generated based on master key:MKicv_bpt which generates the ICV key of BPT beforehand stored in the device, initial value:IVicv_bpt when generating the ICV key of BPT, and Media ID. Specifically based on alteration check value (ICV) generation key Kicv_bpt=DES (E, MKicv_bpt, media 1 ID^IVicv_bpt), it is generated. The semantics of a formula means that encryption processing by DES Mohd by master key:MKicv_bpt is performed to the exclusive OR of media 1ID and initial value (IVicv_bpt).
[0132] Next, a memory interface generates ICV' of BPT using generated alteration check value (ICV) generation key Kicv_bpt, and performs collating processing (ICV'=ICV?) with the right ICV value beforehand stored in BPT (S230). In addition, generation processing of ICV' is performed by the processing which applied generated alteration check value (ICV) generation key Kicv_bpt using initial value IVbpt based on DES Mohd who explained by above-mentioned drawing 14 . In addition, ICV stored as incidental information on BPT is generated based on the data containing Media ID, and the check of ICV has the function which has not only the existence of a data alteration of BPT but verification of not being BPT with a just media proper, i.e., BPT copied to other media.
[0133] When it is ICV'=ICV (it is Yes at S231), it is judged with it being a just thing without the alteration in which BPT was stored in just media, and sets to the condition which can be referred to in the cases, such as contents processing, and the one set flag of media is set to 1 (set) (S232). When it is ICV'!=ICV (it is No at S231), it is judged with those with an alteration by BPT, the contents processing based on reference

processing of BPT is forbidden, and processing is ended. A busy flag is set to 0 by termination of the above processing (S233).

[0134] On the other hand, a control-section side transmits a status read-out command to a memory interface (S234), and the one set flag of media is saved for the busy flag having been set to 0 as conditions (it is Yes at S235) (S236). In the case of 1 which shows that media 1 were set effectively, and others, it is set to 0 when judged with the one set flag of media saved not having the alteration of BPT.

[0135] Next, the media 2 check processing at the time of a device being equipped with media 2 is explained using drawing 20 and drawing 21 . Media 2 are media which perform mutual recognition with a device, as explained using drawing 2 .

[0136] From step S301 of drawing 20 , since the step of S304 is the same as steps S201-S204 in check processing of media 1, explanation is omitted.

[0137] In step S305, a memory interface performs mutual recognition processing with media 2.

[0138] The processing sequence of the mutual recognition approach (ISO/IEC 9798-2) which used the common key encryption system for drawing 22 is shown. In drawing 22 , although DES is used as a common key encryption system, if it is a common key encryption system, other methods are possible. In drawing 22 , the random number Rb whose B is 64 bits is generated first, and ID (b) which is Rb and self ID is transmitted to A. A which received this newly generates the 64-bit random number Ra, in order of Ra, Rb, and ID (b), Key Kab is used for it by CBC Mohd of DES, it enciphers data, and returns them to B. In addition, Keys Kab are a private key common to A and B, and an authentication key. The encryption processing with the key Kab using CBC Mohd of DES For example, in the processing using DES, carry out the exclusive OR of initial value and Ra, and it sets in the DES encryption section. Encipher using Key Kab, generate a cipher E1, carry out the exclusive OR of the ciphers E1 and Rb continuously, and it sets in the DES encryption section. It enciphers using Key Kab, a cipher E2 is generated, the exclusive OR of the ciphers E2 and ID (b) is carried out further, and the cipher E3 which enciphered using Key Kab and was generated generates transmit data (Token-AB) in the DES encryption section.

[0139] B which received this decrypts received data with the key Kab (authentication key) too stored in each record component as a common private key. First, the decryption approach of received data decrypts a cipher E1 with the authentication key Kab, it carries out an exclusive OR to initial value, and it obtains a random number Ra. Next, a cipher E2 is decrypted with the authentication key Kab, the exclusive OR

of E1 is carried out to the result, and Rb is obtained. Finally, a cipher E3 is decrypted by the authentication key Kab, the exclusive OR of E2 is carried out to the result, and ID (b) is obtained. In this way, Rb and ID (b) verify whether it is in agreement with what B transmitted among Ra, Rb(s), and ID (b) which were obtained. When it passes in this verification, B attests A as a just thing.
[0140] Next, B generates the session key (Kses) used after authentication with a random number. And in order of Rb, Ra, and Kses, by CBC Mohd of DES, the authentication key Kab is used, it enciphers, and A is returned.
[0141] A which received this decrypts received data by the authentication key Kake. The decryption approach of received data is the same as that of decryption processing of B. In this way, Rb and Ra verify whether it is in agreement with what A transmitted among Rb(s), Ra, and Kses(es) which were obtained. When it passes in this verification, A attests B as a just thing. After attesting the partner of each other, the session key Kses is used as a common key for the secret communication after authentication.
[0142] In addition, in the case of verification of received data, when injustice and an inequality are found, subsequent mutual data communication processing is forbidden as that in which mutual recognition failed.
[0143] The device of this invention, the mutual recognition between media, and a key (session key) share processing flow are shown in drawing 23 and drawing 24 . In drawing 23 and drawing 24 , it is processing [in / left-hand side, and / in right-hand side / the controller of media 2]. [the memory interface of a device]
[0144] First, media 2 controller generates a random number Ra (S401), and media 2ID which is Ra and self ID is transmitted to a device memory interface (S402). The device memory interface which received this (S403) applies master key:MKake for authentication key generation which self owns to media 2ID which received, and the exclusive OR of initial value (IV_ake), performs DES encryption processing, and generates the authentication key Kake (S404). Furthermore, a device memory interface newly generates a random number Rb (S405). Carry out the exclusive OR of initial value IV_auth and the Rb, and it enciphers using Key Kake. Generate a cipher E1 and the exclusive OR of the ciphers E1 and Ra is carried out continuously. Encipher using Key Kake, generate a cipher E2, and the exclusive OR of a cipher E2 and media 2ID is carried out further. It enciphers using Key Kake and data E1||E2||E3 which generated and (S406) generated the cipher E3 are transmitted to media 2 controller (S407). [||] means association of data.
[0145] Media 2 controller which received this (S408) decrypts received data with the

authentication key Kake (S409). First, the decryption approach of received data decrypts a cipher E1 with the authentication key Kake, it carries out an exclusive OR to initial value, and it obtains random-number Rb'. Next, a cipher E2 is decoded with the authentication key Kake, the exclusive OR of E1 is carried out to the result, and Ra' is obtained. Finally, a cipher E3 is decoded with the authentication key Kake, the exclusive OR of E2 is carried out to the result, and media 2ID' is obtained. in this way -- obtaining -- having had -- Ra -- ' -- Rb -- ' -- media -- two -- ID -- ' -- inside -- Ra -- ' -- and -- media -- two -- ID -- ' -- media -- two -- having transmitted -- a thing -- being in agreement -- or -- verification (S410, S411) -- carrying out . When it passes in this verification, media 2 attest a device as a just thing. Mutual recognition should go wrong (S413) and Ra' and media 2ID' stop subsequent data communication, when inharmonious, transmit data and.

[0146] Next, media 2 controller generates the random number as a session key (Kses) used after authentication (S412). Next, in step S421 of drawing 24 , in order of Ra, Rb, and Kses, by CBC Mohd of DES, the authentication key Kake is used, and it enciphers, and transmits to a device memory interface (S422).

[0147] The device memory interface which received this (S423) decodes received data with the authentication key Kake (S424). in this way -- obtaining -- having had -- Ra -- ″ -- Rb -- ″ -- Kses -- inside -- Ra -- ″ -- and -- Rb -- ″ -- a device -- having transmitted -- a thing -- being in agreement -- or -- verification (S425, S426) -- carrying out . When it passes in this verification, a device attests media 2 as a just thing (S427). After attesting the partner of each other, the session key Kses is shared (S429) and it is used as a common key for the secret communication after authentication. Mutual recognition should go wrong (S428) and Ra″ and Rb″ stop subsequent data communication, when inharmonious, transmit data and.

[0148] Explanation is continued about recognition processing of return and media 2 to drawing 20 . In step S305, above-mentioned mutual recognition and key share processing are performed, and if it is checked that mutual recognition has been successful at step S306, comparison collating with ID of the media 2 which received at the time of mutual recognition processing, and the list of the RIBOKU (abatement) media 2 under already set RIBOKESHON list will be performed (S307).

[0149] When it does not exist while ID which is in agreement with Reception ID listed, if the wearing media 2 are not the media for RIBOKU but media available to validity, they will be judged (it sets to S308 and is No), they set the media 2 effective flag of a status register to 1 (effective) (S309), and set a busy flag to 0 (standby) (S310). When ID which is in agreement with Reception ID is during a RIBOKESHON list (it sets to

S308 and is Yes), the wearing media 2 are the media for RIBOKU, they judge with the ability not to use effectively, and do not perform validation processing of the effective flag of step S309, but set a busy flag to 0 (standby) at step S310, and end processing.
[0150] On the other hand, in step S311, a control section transmits a status read-out command to a memory interface, and a media flag condition is checked, only when it is validity (flag: 1) (it is Yes at S313), processing is continued, and after checking that the busy flag has been set to 0 (standby) (S312), when it is an invalid (flag: 0) (it is No at S313), it ends processing.
[0151] Next, it progresses to drawing 21 and a control section transmits the file quota table call command about media 2 to a memory interface (S321), and a memory interface transmits the sector read-out command with which the file quota table was stored to media 2 (S322), it receives a file quota table from media 2, and transmits it to a control section (S323).
[0152] If the file quota table corresponding to the data stored in media 2 is received (S324), a control section will perform read-out processing of a block permission table (BPT) based on a table (S325), and will transmit the set command of BPT, and BPT to a memory interface (S326). Set processing of BPT will judge whether elimination for every block is possible with reference to BPT in the case of contents processings, such as contents write-in processing from media, if it performs only when BPT is effective, and BPT is set. The latter part explains data write-in processing in which actual BPT was referred to.
[0153] If the set command of a block permission table (BPT) and BPT are received from a control section (S327), a memory interface will set the busy flag of a status register to 1 (busy) (S328), and will generate alteration check value (ICV) generation key Kicv_bpt for the alteration check of BPT (S329).
[0154] Alteration check value (ICV) generation key Kicv_bpt for the alteration check of BPT is generated based on master key:MKicv_bpt which generates the ICV key of BPT beforehand stored in the device, initial value:IVicv_bpt when generating the ICV key of BPT, and media 2ID. Specifically based on alteration check value (ICV) generation key Kicv_bpt=DES (E, MKicv_bpt, - Dear 2 ID^IVicv_bpt), it is generated. The semantics of a formula means that encryption processing by DES Mohd by master key:MKicv_bpt is performed to the exclusive OR of media 2ID and initial value (IVicv_bpt).
[0155] Next, a memory interface generates ICV' of BPT using alteration check value (ICV) generation key Kicv_bpt and IVbpt which were generated, and performs collating processing (ICV'=ICV?) with the right ICV value beforehand stored in BPT (S330). In

addition, generation processing of ICV' is performed by the processing which applied generated alteration check value (ICV) generation key Kicv_bpt using initial value IVbpt based on DES Mohd who explained by above-mentioned drawing 14 . In addition, ICV stored as incidental information on BPT is generated based on the data containing media 2ID, and the check of ICV has the function which has not only the existence of a data alteration of BPT but verification of not being BPT with a just media proper, i.e., BPT copied to other media.
[0156] When it is ICV'=ICV (it is Yes at S331), it is judged with it being a just thing without the alteration in which BPT was stored in just media, and sets to the condition which can be referred to in the cases, such as contents processing, and the two set flag of media is set to 1 (set) (S332). When it is ICV'!=ICV (it is No at S331), it is judged with those with an alteration by BPT, the contents processing based on reference processing of BPT is forbidden, and processing is ended. A busy flag is set to 0 by termination of the above processing (S333).
[0157] On the other hand, a control-section side transmits a status read-out command to a memory interface (S334), and the two set flag of media is saved for the busy flag having been set to 0 as conditions (it is Yes at S335) (S336). In the case of 1 which shows that media 2 were set effectively, and others, it is set to 0 when judged with the two set flag of media saved not having the alteration of BPT.
[0158] (Data file read-out processing) Next, read-out processing of a data file is explained using the flow of drawing 25 . Contents data files, such as music data and image data, and the RIBOKESHON list further mentioned above are also included in a data file. The flow shown in drawing 25 is a processing flow common to read-out of the data file stored in an internal memory or external memory (media 1, media 2). In drawing 25 , left-hand side is the control section of a device, and right-hand side is processing of the memory interface of a device.
[0159] first, reading appearance of the control section is carried out from a file quota table (refer to drawing 17), it carries out sector S (i) reading appearance which acquired the sector address (S (1) -S (k)) of object data (S501), and was acquired to the memory interface, and carries out sequential transmission (S502, S503) of the command. If a sector S(i) read-out command is received (S504), a memory interface Set a busy flag as 1 (busy) (S505), judge whether receiving sector S (i) is an internal memory and external memory (S506), and when it is external memory Judge (S507), and when a set flag is 1, whether the set flag of media 1 or media 2 is 1 (it is shown that media are set effectively) furthermore with reference to a block permission table (BPT), BPT judges whether reading appearance was carried out, reading appearance

of the sector S (i) which is an object was carried out, and it has set up as a block for authorization (S508). When BPT has setting out of a read-out authorization block, the data of an applicable sector are read from external memory (S509).
[0160] In addition, when the data for read-out are data in the internal memory by which management by BPT is not made, steps S507 and S508 are skipped. When the judgment of steps S507 and S508 is No (i.e., when the set flag of the media which stored sector S (i) is not 1), or when read-out authorization of sector S (i) is not set as BPT, it progresses to step S513, and reads as a read-out error, and a success flag is set to 0.
[0161] In the judgment block of steps S506-S508, if read-out of object sector S (i) is judged as activation being possible Reading appearance of the applicable sector is carried out from memory, and error correction processing based on the error correcting code of the redundancy section set up corresponding to the sector is performed (S510). What (S511) the error correction was [the thing] successful is checked, a read-out success flag is set to 1 (success), a read-out result is stored in a buffer (S512), and a busy flag is set as 0 (standby) (S513). When an error correction goes wrong, a read-out success flag is set as 0 (failure) (S513), and processing is ended.
[0162] Moreover, in steps S515-S520, a control section reads the status of a memory interface and a busy flag sets it in the condition of 0. Read a condition [a read-out success flag being 1], pick out data from a buffer, save them, and sequential increment of the address is carried out. After repeating and performing processing which picks out data from a buffer one by one, and saves them and saving all the sectors for read-out, a file is constituted from all read-out sector data, and processing is ended.
[0163] (File write-in processing) Next, write-in processing of a data file is explained using the flow of drawing 26 . The flow shown in drawing 26 is a common processing flow at the time of writing a file in an internal memory or external memory (media 1, media 2). In drawing 26 , left-hand side is the control section of a device, and right-hand side is processing of the memory interface of a device.
[0164] First, a control section divides the file for writing into a sector. The divided data are set to D (1) - D (k). A control section sets up write-in sector [of each data D (i)] S (i) next, and carries out sequential transmission (S602-S604) of the data D (i) to a sector S (i) write command at a memory interface. If a sector S (i) write command is received (S605), a memory interface Set a busy flag as 1 (busy) (S606), judge whether receiving sector S (i) is an internal memory and external memory (S607), and

when it is external memory Judge (S608), and when a set flag is 1, whether the set flag of media 1 or media 2 is 1 (it is shown that media are set effectively) Furthermore with reference to a block permission table (BPT), it judges whether BPT wrote in and sector S (i) which is an object is set up as a block for write-in authorization (S609). When BPT has setting out of a write-in authorization block, the error correcting code set up corresponding to a sector is generated (S610), the redundancy section which has Data D (i) and an error correcting code in sector S (i) is written in, a write-in success flag is set to 1 (success), and a busy flag is set as 0 (standby) (S614).
[0165] In addition, when the data for writing are write-in processing into the internal memory by which management by BPT is not made, steps S608 and S609 are skipped. When the judgment of steps S608 and S609 is No (i.e., when the set flag of media is not 1), or when write-in authorization of sector S (i) is not set as BPT, it progresses to step S613, and writes in as a write error, and a success flag is set to 0.
[0166] moreover, a control section carries out reading appearance of the status of a memory interface in steps S616-S620, and in the condition of 0, a busy flag carries out sequential increment of the address a condition [a write-in success flag being 1], and transmits write-in data to a memory interface one by one. After all processings are completed, an update process of a file quota table is performed (S621), the updated file quota table is transmitted to a memory interface with an updating command (S622), and a memory interface performs write-in processing of a file quota table according to a command (S623).
[0167] [Encryption processing which applied the encryption key according to a sector location], next the encryption processing which applied the encryption key according to a sector location are explained. In order to protect copyright etc., encryption to the contents section may be performed, but when it enciphers using one encryption key to the whole contents section, the cipher of the large quantity under the same key occurs, and there is a danger that an attack will become easy. Usually, the contents section is divided as much as possible, and it can say that it is more desirable to encipher each with a different key. As a smallest unit of contents encryption by this system, although a sector is mentioned, in the object of saving a key to a header field, only the number of sectors makes the data areas of the memory area which the size of a header became huge and was restricted since 8 bytes (in the case of DES) or 16 bytes (Triple DES (in the case of Triple-DES)) of key information was needed decrease in number, and is not desirable practically. Moreover, although header size will not be affected if the approach of storing the key for enciphering the sector in a part for the data division of each sector is taken, in the case of a system which has a

file system by the control-section side, large modification should be needed at the file system itself with data size losing in weight, since it becomes impossible to put data on the field of a key.
[0168] then, the inside of the security header (refer to drawing 7) which is the attribute information on each contents previously explained by the system of this invention -- for example, several sectors per block of media -- the key information on M individual corresponding to M is stored, and these are applied as an encryption key to each sector (refer to drawing 8). Kc_Encrypted0 in the security header shown in drawing 7 - Kc_Encrypted31 show 32 encryption keys Kc. In addition, [Encrypted] shows that each key Kc is enciphered and stored. It considered as the configuration which chooses a key with the location within a block of a sector from two or more of these keys, and is used as an encryption key corresponding to a sector.
[0169] Drawing explaining a response with the key storing configuration in the security header generated as header information of contents by drawing 27 corresponding to contents, each storing key, and each sector in the memory set as the application object of each key is shown. It is drawing having simplified and shown the key storing configuration in the security header which drawing 27 (a) explained using drawing 7 previously. M keys (contents key) to Kc(0) -Kc (M-1) are stored in the security header of drawing 27 (a). Various information, such as a version and a contents type, is stored in a header besides a key, and ICV further for the alteration check of header information is stored.
[0170] This M contents key is used for the data encryption which each is matched with each sector and stores in each sector as shown in drawing 27 (b). As previously explained using drawing 3 , as the flash memory which performs elimination in a block unit is shown in drawing 27 (b), a data storage field is divided per block, and each block is further divided into two or more sectors. For example, Key Kc (0) is applied as a data encryption key stored in the sector 0 of each block of memory, and it considers as the data encryption key which stores Key Kc (s) in the sector s of each block of memory. Furthermore, it applies as a data encryption key which stores Key Kc (M-1) in the sector M-1 of each block of memory.
[0171] Thus, the security of storing data (ex. contents) is raised by storing data with the application of a different cryptographic key corresponding to a sector. That is, when the whole contents are enciphered with one key, it is because it is impossible to decode the whole data by leakage of one key to decode of the whole contents by key leakage being attained according to this configuration.
[0172] The single DES to which encryption algorithm performs DES encryption

processing by one cryptographic key is applied. Moreover, it is good also as an encryption configuration which applied not the single DES but the Triple DES (Triple DES) which uses two or more keys for encryption.

[0173] The example of a detail configuration of Triple DES (Triple DES) is shown in drawing 28 . As shown in drawing 28 (a) and (b), there are two following different modes in the configuration as Triple DES (Triple DES) typically. Drawing 28 (a) shows the example which used two cryptographic keys, and processes in order of the encryption processing with a key 1, the decryption processing with a key 2, and encryption processing according to a key 1 further. A key is used two kinds in order of K1, K2, and K1. Drawing 28 (b) shows the example which used three cryptographic keys, processes in order of the encryption processing with a key 1, the encryption processing with a key 2, and encryption processing according to a key 3 further, and performs encryption processing 3 times. Three kinds of keys are used for a key in order of K1, K2, and K3. Thus, it is possible to raise security reinforcement by considering as the configuration which two or more processings are made to follow as compared with Single DES.

[0174] The example of a configuration which performed encryption processing by Triple DES to drawing 29 with the application of the pair of two different cryptographic keys for every sector of the data stored in memory is shown. As shown in drawing 29 , the sector 0 of each block performs Triple DES encryption using Key Kc (0) and two keys of Kc (1), Sector s performs Triple DES encryption using Key Kc (s) and two keys of Kc (s+1), and a sector M−1 performs Triple DES encryption using two keys of Keys Kc (M−1) and Kc (0). Even in this case, the number of keys stored in a header is M pieces, does not need to make the number of key storing shown by drawing 27 (a) increase, and becomes possible [raising security].

[0175] Furthermore, the example of a data encryption configuration in a mode which is different in drawing 30 is shown. Drawing 30 is the mode which performed Triple DES encryption for two continuous sector fields of each block of memory, using two keys as one encryption block. As shown in drawing 30 , the sector 0 and sector 1 of each block Triple DES encryption is performed using Key Kc (0) and two keys of Kc (1). Sector 2s and sector 2s+1 Performing Triple DES encryption using two keys of Keys Kc (2s) and Kc (2s+1), a sector M−2 and a sector M−1 perform Triple DES encryption using two keys of Keys Kc (M−2) and Kc (M−1). Thus, processing relief of an encryption process or a decode process can be enabled by applying the same encryption processing to two or more sectors.

[0176] Various configurations as a configuration which performs encryption for every

sector using the key which stored two or more keys in the header other than the example shown in drawing 27 , drawing 29 , and drawing 30 , and was chosen from the two or more keys are possible. For example, although considered as the configuration which stores the key of the number of sectors, and the same number in a header in drawing 27 , and 29 and 30, when the number of sectors is M, for example, it is good [a sector 0 and Sector s] also as a configuration of enciphering with the same key, using the number of storing keys as N (N<M). Moreover, it is good also as a configuration which applies the Triple DES by two or more key sets which set the number of storing keys to L (L>M), and are completely different for every sector.
[0177] [The addition configuration of the alteration check value (ICV) of a sector unit], next the addition configuration of the alteration check value (ICV) of a sector unit are explained. About the data constituted ranging over two or more sectors, when checking the justification, generally it was common to have considered as the configuration to which the alteration check value (ICV) mentioned above at the last of the whole contents data etc. is made to add. In the addition configuration of ICV of such whole data, it is each sector unit which constitutes data, and justification cannot be checked.
[0178] Moreover, if ICV is put into the storing field and this field of contents which are live data when it stores ICV, the fields which can be used as the part data division will decrease in number. If ICV for every sector is put into each sector to the data in a sector, in order that the file system of a device may perform processing which reads data per data division The processing which removes ICV in the sector in the read data division the processing for separating and taking out the data actually used from ICV, i.e., once, It is necessary to perform processing which connects the data in the taken-out sector with two or more sectors, and it necessary to newly build the file system for performing the processing. Furthermore, if these ICV checks are performed by the control section, the load of the part of the processing will be applied to a control section.
[0179] In the data processor of this invention, in order to enable a data alteration check for every sector, ICV was set up for every sector and the ICV setting-out location was made into the redundancy section field beforehand set up as a field which is not read by the file system of a device instead of a live-data field. By considering as the configuration which puts ICV on the redundancy section, it becomes unnecessary to place ICV into data, and many fields of data division can use. Moreover, by putting ICV on the redundancy section, since carving and data connection processing of data division and ICV become unnecessary, the continuity of

data read-out is maintained.
[0180] When reading data, ICV check processing for every sector is performed in the memory interface section 300 (refer to drawing 2), it is judged with those with an alteration, and when it is invalid data, the transfer to a control section 205 (refer to drawing 2) is not performed. Moreover, at the time of data writing, ICV of each sector is calculated in the memory interface section 300, and processing written in the redundancy section is performed.
[0181] In addition, with each sector, it describes whether ICV is added or it does not carry out to a security header (Security Header), and specifies. About this configuration, as shown during explanation of the security header configuration of drawing 7 , the ICV flag in a security header (ICV Flag) has a flag for the number of sectors (32 sectors) within a block, and ICV addition and un-adding are shown. [of each sector within a block] For example, it considers as 0:ICV nothing and those with 1:ICV, and is set up.
[0182] The data utilization section and the redundancy section configuration of each sector are shown in drawing 31 . Like drawing 31 (a), the data stored in memory (flash memory) are divided and stored in a block unit field with two or more sector fields. As shown in (b), each sector is constituted by the redundancy section which stored information with which it is read as live data (ex. contents) by the file system of a device, such as 512 or 1024 bytes of data utilization section, and ECC (Error Correction Code) that is not read depending on a file system, for example.
[0183] The capacity of this redundancy section is 16 bytes or 20 bytes of field decided beforehand, and the file system of a device recognizes this redundancy section as a non-data area, and does not read it in data (contents) reading processing. Generally, ECC stored in the redundancy section does not use the whole redundancy section, but a non-using field (reserve field) exists in the redundancy section. The alteration check value (ICV) of each sector is stored in this reserve field.
[0184] As shown in drawing 31 (c), the conventional data connection processing of only connecting the data division in which only what is purely used as data was stored, and the same processing of connection processing of the data division by the file system of the device at the time of storing ICV in the redundancy section are attained. Therefore, new processing is not needed at all that the file system of a device should just only connect the data-division field except the redundancy section.
[0185] The justification of data is verifiable with this configuration per sector of the data which consist of two or more sectors. Moreover, the data area which can be used for data is utilizable as it is by putting ICV for an alteration check into the redundancy

section. Moreover, only the right sector judged to be the right (with no alteration) is transmitted to a control section as a result of an ICV check. Moreover, since an ICV check is performed in the memory interface section, there is effectiveness of the burden of a control section not being placed.
[0186] [Preservation processing of a contents key with the individual key in media], next the preservation processing configuration of a contents key with the individual key in media are explained. Previously, as explained using drawing 7 , two or more contents keys (Kc_Encryptedxx) as a cryptographic key of a sector response and a contents check value generation key (Kicv_Encrypted) are enciphered and stored in the security header constituted corresponding to contents.
[0187] One mode of encryption of these keys has the configuration enciphered and stored with the delivery key Kdist beforehand stored in the memory section 321 (refer to drawing 4) of the memory interface of a device. For example, it is Kc_Encrypted0=Enc (Kdist, Kc (0)). Here, it is shown that Enc (a, b) is data which enciphered b by a. Thus, the configuration which enciphers using the delivery key Kdist of a device and stores each key in a security header is one configuration.
[0188] Furthermore, it has media 2, i.e., the cipher-processing section, and there is a mode which enciphers the contents key about the contents stored in media 2 using the proper key of media 2 and an ICV generation key in the media which perform mutual recognition with a device and perform contents processing. Hereafter, the proper key of media 2 and here explain the processing which stores in a security header the contents key and contents ICV generation key which were enciphered using the media 2 preservation key Ksto.
[0189] The media 2 preservation key Ksto is stored in the internal memory 235 of media 2 controller 231 of media 2,230 as shown in drawing 2 . Therefore, encryption processing of the contents key which used the media 2 preservation key Ksto, and an ICV generation key, and decode processing are performed by the media 2 side. When the device equipped with media 2 performs acquisition or storing processing to a security header for a contents key and an ICV generation key on the occasion of contents utilization of media 2, it is necessary to perform encryption of a key, and decode processing by the media 2 side. In the data processor of this invention, it made it possible to process these by CBC (Cipher Block Chaining) Mohd.
[0190] The encryption processing configuration of the key in CBC Mohd is shown in drawing 32 . This encryption processing is performed in the cipher-processing section 236 (refer to drawing 2) of media 2. Exclusive OR of initial value IV_keys stored in the internal memory 235 and contents check value generation key Kicv_cont is performed,

DES encryption which applied the preservation key Ksto in which the result was stored by the internal memory 235 of media 2 is performed, and it stores in a header by making the result into Kicv_cont Encrypted. Furthermore, exclusive OR of Kicv_cont Encrypted and the contents key Kc (0) corresponding to a sector corresponding to a sector (0) is performed, DES encryption which applied the preservation key Ksto in which the result was stored by the internal memory 235 of media 2 is performed, and it considers as one encryption contents key stored in a header by setting the result to Kc(0) Encrypted. Furthermore, exclusive OR of Kc(0) Encrypted and the contents key Kc (1) corresponding to a sector corresponding to a sector (1) is performed, DES encryption which applied the preservation key Ksto to the result is performed, and the result is set to Kc(1) Encrypted. Hereafter, these processings are repeated and performed and it considers as the key data for header storing.
[0191] Next, the decode processing configuration of the key in CBC Mohd is shown in drawing 33 . This decode processing is performed in the cipher-processing section 236 (refer to drawing 2) of media 2. First, the contents key Kc (0) corresponding to a sector corresponding to a sector (0) is outputted by performing DES decode processing which applied the preservation key Ksto stored in the internal memory 235 of media 2 to Kc(0) Encrypted, and carrying out exclusive OR to initial value IV_keys in which the result was stored by the internal memory 235. Furthermore, the contents key Kc (1) corresponding to a sector corresponding to a sector (1) is outputted by performing DES decode processing which applied the preservation key Ksto to Kc(1) Encrypted, and carrying out exclusive OR of the result to the contents key (0) Kc Encrypted. Hereafter, these processings are repeated and performed and a contents key is acquired. In addition, although the example which used only the contents key as output data is shown in drawing, generation of a contents alteration check value generation key (Kicv_Encrypted) to the contents alteration check value generation key (Kicv) which could apply the same processing and was enciphered also about the contents alteration check value generation key (Kicv_Encrypted) is possible.
[0192] In many cases, encryption of the above-mentioned contents key Kc corresponding to a sector (xx) or a contents alteration check value generation key (Kicv) and decode processing are performed based on the command from the device equipped with media 2. In this case, between a device and media 2, mutual recognition mentioned above will be performed, various processings of contents playback, storing, etc. will be performed a condition [mutual recognition processing having been materialized], and decode of an above-mentioned contents key and encryption

processing will be performed as one of the contents processings of that single string. When transmitting the decoded key (ex. contents key Kc (xx)) between a device and media 2, it is enciphered by the session key Kses generated at the time of mutual recognition. Encryption by this session key Kses and decode processing also become possible [raising security more] by applying CBC Mohd.

[0193] The key stored in the security header in media 2 at drawing 34 is decoded by DES-CBC Mohd, and the processing configuration which enciphers the decoded key data by DES-CBC Mohd with the application of the session key Kses further is shown. The upper case of drawing 34 is the same configuration as drawing 33 , it inputs the enciphered contents key which was taken out from the security header into the DES decode section one by one, performs decode processing with the application of the preservation key Ksto of media 2, and exclusive-OR-acquires the contents key as an output by using an output as the data before initial value or an input data train.

[0194] Encryption processing by DES-CBC Mohd who applied the session key Kses which generated these outputted results further at the time of mutual recognition with a device is performed. SE [which was obtained as a result]0 - SEM-1:Kc(0) Encrypted-Kc(M-1) Encrypted is transmitted to a device. In a device side, the session key Kses generated at the time of mutual recognition with media 2 can be applied about data stream Kc(0) Encrypted-Kc(M-1) Encrypted which received, and contents key K (c) can be acquired by performing decode processing by the same DES-CBC Mohd as drawing 33 . In addition, although the example which used only the contents key as processed data is shown in drawing, it is possible to constitute as processed data similarly about a contents alteration check value generation key (Kicv_Encrypted).

[0195] The detail of the read-out processing from the media of the enciphered data is explained using the flow below [read-out processing of encryption data] drawing 35 . In addition, a data encryption mode has the mode enciphered with a different key for every sector, and the mode which enciphered the whole contents with one encryption key, as mentioned above, and these are judged based on the information on a header. In the flow of drawing 35 , left-hand side is the control section of a device, and right-hand side is processing of the memory interface of a device.

[0196] A control section reads the header file of the contents used as a read-out object first (S701). This processing is performed as processing according to the file read-out processing flow of above-mentioned drawing 25 . Next, the header file read with the header set command is transmitted to a memory interface (S702).

[0197] If a header set command is received (S703), a memory interface will set a busy flag to 1 (busy) (S704), and will verify the alteration check value (ICV) of a header

(S705). In ICV generation processing in which it explained using drawing 14 previously, the ICV check of a header inputs the configuration data of a header as security header verification value generation key Kicv_sh with the application of initial value IVsh, and is performed by processing which collates ICV' which generated and generated ICV', and ICV beforehand stored in the header.
[0198] If judged with having no header alteration by verification (S706), it will be confirmed whether the effective RIBOKESHON list version in a header is 0 (S707). For example, when the contents generated and stored with the self-device are stored in memory, activation of the processing which set the RIBOKESHON list version to 0 and considered the RIBOKESHON list as refer to un-on the occasions, such as regeneration, is enabled.
[0199] When a RIBOKESHON list version is 0, since it is not necessary to refer to a RIBOKESHON list, it progresses to step S710. a version -- un--- when it is 0, the RIBOKESHON list by which the current set is carried out confirms whether it is older than the version of a header (S708), when old, it progresses to S713, a header set success flag is set as 0 (NG), and processing is ended. If the RIBOKESHON list set is not older than the version of a header, it progresses to step S709 and judges whether there is any content ID for read-out with reference to a RIBOKESHON list. When it is, processing is ended as processing which forbids read-out, using a header set success flag as 0 (NG) at step S713.
[0200] If it reads to a RIBOKESHON list and object content ID is not recorded, it progresses step S710 and the contents key Kc enciphered based on header information and contents check value generation key Kicv_cont are decoded. In addition, as the RIBOKESHON list was explained in the flow at the time of starting of previous drawing 16 , it is set up in a memory interface at the time of starting, and after a setup is the RIBOKESHON list made continuously available in the memory interface in the time of playback of contents at the time of wearing of media.
[0201] Previously, as explained using [else / drawing 7], in the security header, two or more contents key Kc(0) -Kc(s) (M-1) as a cryptographic key applied for every above-mentioned sector are enciphered and stored. Moreover, contents check value generation key Kicv_cont for generating the alteration check value (ICV) of contents is also enciphered and stored.
[0202] In advance of decode of contents, the processing which decodes these contents check value generation key Kicv_cont, and performs the alteration check of contents is required, and the processing which decodes contents key Kc(0) -Kc (M-1) is needed.

[0203] The contents key Kc enciphered by drawing 37 and the decode processing flow of contents check value generation key Kicv_cont are shown. Each step of drawing 37 is explained. Processing of drawing 37 is processing in the memory interface of a device. It performs in the cipher-processing section 320 of drawing 4 .

[0204] First, encryption contents check value generation key Kicv_cont is made applicable to decode, and is selected (S801), next setting out of the encryption format type field of a header judges whether it is 0 (S802). When it is the data configuration which made the whole contents one encryption mode irrespective of the sector when an encryption format was 0 and setting out of the encryption format type field is 1, it is an approach using the encryption key of a sector unit explained by above-mentioned drawing 27 etc. When it is an approach using the encryption key of a sector unit, it progresses to step S803 and the encryption contents key (Kc_Encrypted 0-31) set up for every sector is made applicable to decode.

[0205] When judged with an encryption format being 0 at step S802, at step S804, further, the encryption algorithm field of a header is checked and it judges whether 1 (Triple DES) is 0 (single DES). When it is Single DES, only one encryption contents key (Kc_Encrypted0) is added as an object for decode at step S805, and when it is Triple DES, two or more encryption contents keys (Kc_Encrypted 0 and 1) at step S806 are added as an object for decode.

[0206] Next, in step S807, setting out of the contents type field of a header is checked, when setting out is not 2 or 3 (storing contents of media 2), it is step S808 and the data for decode, i.e., encryption contents check value generation key Kicv_cont, and one or more contents keys are decoded with the delivery key Kdist stored in the memory section 321 (refer to drawing 4).

[0207] When setting out is 2 or 3 (storing contents of media 2), the data for decode, i.e., encryption contents check value generation key Kicv_cont, and one or more contents keys are decoded with the preservation key Ksto of media 2 (CBC Mohd) at step S809. The detail of this decode processing is as having explained using drawing 32 , drawing 33 , and drawing 34 .

[0208] Decode processing of one or more contents keys Kc is explained to be encryption contents check value generation key Kicv_cont with the preservation key of the media 2 in step S809 using the flow of drawing 38 . The flow of drawing 38 shows the memory interface of a device to left-hand side, and shows processing of the controller (refer to drawing 2) of media 2 to right-hand side.

[0209] First, a memory interface sets up data for decode K (0) - K (n-1) (encryption contents check value generation key Kicv_cont and one or more contents keys)

(S1001), the CBC decode initialization command is transmitted to media 2 controller (S1003), and media 2 controller sets IVKeys to a register (S1005). Then, a memory interface carries out sequential transmission (S1004) of each key, and media 2 controller receives data for decode K (i) (S1005).
[0210] Next, to data for decode K (i) which received, media 2 controller performs decode processing by CBC Mohd using the preservation key Ksto of media 2 (S1007), and acquires the decoded key data (contents key corresponding to a sector of ex. plurality) (S1008). Next, by the session key which generated the decode key data stream at the time of mutual recognition with a device, media 2 controller performs encryption processing by CBC Mohd, generates data stream K' (i), and transmits a result to a device (S1009). Processing of steps S1007-S1009 is performed based on processing by DES-CBC Mohd of drawing 34 who explained previously.
[0211] The memory interface of a device transmits the CBC quit command to media 2 controller after checking having received K' (i) one by one and having received all data. Media 2 controller clears a register by reception of the CBC quit command (S1014).
[0212] The memory interface of a device decodes K' (i) which received from media 2 by CBC Mohd with the application of the session key Kses generated at the time of mutual recognition with media 2 using initial value IV_keys stored in the memory section 321 (refer to drawing 4) (S1010-S1013, S1015). This decode processing is the same processing as the configuration of drawing 33 explained previously.
[0213] By the above-mentioned processing, a device can decode the enciphered contents key Kc which was stored in the header, and contents check value generation key Kicv_cont, and can acquire each key.
[0214] Next, a continuation of read-out processing of return and an encryption file is explained to drawing 35 . After ending step S710 which is the above-mentioned key decode processing step, it progresses to step S711. At step S711, by using a header as a "read-out header", the memory interface of a device is set as the interior, sets a header set success flag to 1 (success), and carries out 0 (standby) (S714) setting out of the busy flag. On the occasion of contents read-out, processing based on the information on the set-up header is performed.
[0215] On the other hand, a control-section side transmits a status read-out command to a memory interface at step S715, and a busy flag is 0 (standby) (S716), and it progresses to the next processing (drawing 36) a condition [the header set success flag having been set to 1 (success) (S717)].
[0216] in step S721 of drawing 36 , reading appearance of the control section is carried out from a file quota table, and it acquires the sector address (S (1) -S (k)) of

the target contents file, and to a memory interface, one by one, sector S (i) reading appearance of it is carried out, and it transmits a command.
[0217] If a sector S(i) read-out command is received (S724), a memory interface will set a busy flag as 1 (busy) (S725), and will shift to degree step by making for a header success flag to be 1 (success) into conditions (S726). When a header success flag is not 1 (success), processing is ended progressing to step S738 and using a read-out success flag as 0 (NG).
[0218] When a header success flag is 1 (success) Judge (S727), and when it is external memory, whether receiving sector S (i) is an internal memory and external memory Judge (S728), and when a set flag is 1, whether the set flag of media 1 or media 2 is 1 (it is shown that media are set effectively) furthermore with reference to a block permission table (BPT), BPT judges whether reading appearance was carried out, reading appearance of the sector S (i) which is an object was carried out, and it has set up as a block for authorization (S729). When BPT has setting out of a read-out authorization block, the data of an applicable sector are read from external memory (S730).
[0219] In addition, when the data for read-out are data in the internal memory by which management by BPT is not made, steps S728 and S729 are skipped. When the judgment of steps S728 and S729 is No (i.e., when the set flag of the media which stored sector S (i) is not 1), or when read-out authorization of sector S (i) is not set as BPT, it progresses to step S738, and reads as a read-out error, and a success flag is set to 0.
[0220] In the judgment block of steps S726-S729, if read-out of object sector S (i) is judged as activation being possible, reading appearance of the applicable sector will be carried out from memory, error correction processing based on the error correcting code of the redundancy section set up corresponding to the sector will be performed (S731), and what (S732) the error correction was [the thing] successful will be checked. Next, with reference to the ICV flag (refer to drawing 7) of a header, it judges whether the sector for read-out is a processing object by the alteration check value (ICV). As previously explained using drawing 31 , each sector stores ICV for an alteration check in the redundancy section, and the alteration check in a sector unit is possible for it.
[0221] ICV generation processing of having applied initial value IVcont, having inputted the data for an alteration check (sector data), and having explained it as contents check value generation key Kicv_cont obtained by decode processing of step S710 in step S734 using drawing 14 when it was the object of the alteration check by ICV

performs, and it asks for ICV', collating by ICV stored in the redundancy section of a sector carries out, and if in agreement, it will judge have no alteration.

[0222] If judged with having no alteration by the ICV check, it progresses to step S737, and decode processing of data is performed and read based on header information, a success flag is set as 1 (success), and decode data are stored in a buffer.

[0223] Moreover, in steps S740-S746, a control section reads the status of a memory interface and a busy flag sets it in the condition of 0. Read a condition [a read-out success flag being 1], pick out data from a buffer, save them, and sequential increment of the address is carried out. After repeating and performing processing which picks out data from a buffer one by one, and saves them and saving all the sectors for read-out, a file is constituted from all read-out sector data, and processing is ended.

[0224] The detail of data-division decode processing of step S736 of drawing 36 is explained using drawing 39 . This decode processing is performed in the cipher-processing section 320 (refer to drawing 4) of the memory interface of a device.

[0225] First, the data storage sector location for decode is set to s (0<=s<=31 (in the case with 32 sectors)) (S1101). Next, it is confirmed whether the sector is an object for encryption (S1102). This check is judged based on the encryption flag (Encryption Flag) of a security header (refer to drawing 7). When it is not an object for encryption, decode processing is not performed but processing is ended. When it is an object for encryption, an encryption format type is checked (S1103). This checks encryption format type (Encryption Format Type) setting out in a security header, and it judges whether the whole contents explained by drawing 8 are made into one encryption mode, or encryption processing using a key which is different into each sector is performed.

[0226] When the encryption format type (Encryption Format Type) set point is 0, it is the case where the whole contents are being made into one encryption mode. In this case, encryption algorithm (Encryption Algorithm) is judged in step S1104. Encryption algorithm has set up Single DES and Triple DES (refer to drawing 28), and when judged with it being Single DES, it performs decode processing of encryption contents with the application of one contents key Kc (0) (S1106). When judged with it being Triple DES, decode processing of encryption contents is performed with the application of two contents keys Kc (0) and Kc (1) (S1107).

[0227] It is the case where encryption processing using a key which is different into each sector is being performed at step S1103 on the other hand when the code format type (Encryption FormatType) set point is 1. In this case, encryption algorithm (Encryption Algorithm) is judged in step S1105. Encryption algorithm has set up Single DES and Triple DES (refer to drawing 28), and when judged with it being Single DES, it performs decode processing of encryption contents into each sector with the application of the contents key Kc (s) set up corresponding to each sector (s) (S1108). When judged with it being Triple DES, with the application of two contents keys Kc and Kc (s+1mod32), decode processing of the encryption contents for every sector is performed (s) (S1109).

[0228] The processing mode from which decode processing of sector data differs is shown in drawing 40 . In drawing 40 , steps S1201-S1208 are the same as each steps S1101-S1108 of drawing 39 . Steps S1209-S1211 differ from drawing 39 .

[0229] In step S1205, when judged with encryption algorithm being Triple DES, sector No. (s) is judged in step S1209, when the number of s is odd, renewal of s=s -1 is performed (S1210), and decode processing (S1211) according the key applied to each sector to Triple DES is performed as (s), Kc, and Kc (s+1).

[0230] As mentioned above, it is enciphered and regeneration accompanied by decode processing of the stored data is performed according to a process which was explained using drawing 35 - drawing 40 .

[0231] The detail of the data encryption write-in treatment process to media is explained using the flow below [data encryption write-in processing], next drawing 41 . In addition, a data encryption mode has the mode enciphered with a key which is different for every sector as mentioned above, and the mode which enciphered the whole contents with one encryption key. These are set as header information. In the flow of drawing 41, left-hand side is the control section of a device, and right-hand side is processing of the memory interface of a device.

[0232] A control section first transmits the header generation command corresponding to the storing contents used as a read-out object, and the parameter as header information to a memory interface. (S1301) .

[0233] A memory interface will set a busy flag to 1 (busy), if a header generation command is received (S1302) (S1303), and it judges whether a receiving parameter is in an allowed value (S1304). When it has the parameter range which can be beforehand set as a header and the receiving parameter has exceeded the range which can be set up as compared with the receiving parameter, a memory interface sets a header generation success flag as 0 (NG) in step S1310, and ends processing. When a receiving parameter is in an allowed value, the effective RIBOKESHON list version of a header is set as 0 (S1305), and data processing of referring to [of a RIBOKESHON list] un-is made possible. Setting up an effective RIBOKESHON list version as 0 performs setting out which makes possible data processing (playback) of refer to [of a RIBOKESHON list] un-by the premise that it is guaranteed that they are contents just about the contents which performed storing processing with a self-device.

[0234] In addition, they are the contents which write-in contents received from the outside through means of communications. By storing in a header the RIBOKESHON list version which an identifier is added to receiving contents and should be referred to, if collating with the RIBOKESHON list inside a device is possible Identifier collating processing using the same RIBOKESHON list as steps S707-S709 performed instead of the above-mentioned processing in file decode read-out processing in which drawing 35 was used and explained previously may be performed.

[0235] Next, in step S1306, it is based on header information, and the contents key Kc and contents alteration check value (ICV) generation key Kicv_cont are generated, and it enciphers. The detail of generation and encryption processing is shown for the contents key Kc of step S1306, and contents alteration check value generation key Kicv_cont in drawing 43 . Processing of drawing 43 is performed in the cipher-processing section 320 (refer to drawing 4) of the memory interface of a device. The flow of drawing 43 is explained.

[0236] First, encryption contents check value generation key Kicv_cont is generated based on a random number, and it considers as the object for encryption (S1401), next setting out of the encryption format type field of a header judges whether it is 0 (S1402). When it is the configuration which makes the whole contents one encryption mode irrespective of a sector when an encryption format is 0 and setting out of the encryption format type field is 1, it is an approach using the encryption key of a sector unit explained by above-mentioned drawing 27 etc. When using the encryption key of a sector unit, it progresses to step S1403, the contents key (in Kc(0) -Kc(31), the (case with 32 sectors)) set up for every sector is generated, and it considers as the object for encryption.

[0237] When judged with an encryption format being 0 at step S1404, at step S1404, further, the encryption algorithm field of a header is checked and it judges [1 (Triple DES) or] whether it is 0 (single DES). When it is Single DES, one contents key (Kc (0)) is generated at step S1405, and when it is Triple DES in addition as an object for encryption, two or more contents keys (Kc (0), Kc (1)) at step S1406 are generated, and it adds as an object for encryption.

[0238] Next, in step S1407, setting out of the contents type field of a header is checked, when setting out is not 2 or 3 (storing contents of media 2), it is step S1408

and one or more contents keys are enciphered as data, i.e., contents check value generation key Kicv_cont, with the delivery key Kdist stored in the memory section 321 (refer to drawing 4).
[0239] When setting out is 2 or 3 (storing contents of media 2), one or more contents keys are enciphered as data, i.e., contents check value generation key Kicv_cont, with the preservation key Ksto of media 2 (CBC Mohd) at step S1409. The detail of this encryption processing is as having explained using drawing 32 , drawing 33 , and drawing 34 .
[0240] Encryption processing of one or more contents keys Kc is explained to be contents check value generation key Kicv_cont with the preservation key of the media 2 in step S1409 using the flow of drawing 44 . The flow of drawing 44 shows the memory interface of a device to left-hand side, and shows processing of the controller (refer to drawing 2) of media 2 to right-hand side.
[0241] First, the memory interface by the side of a device Data for encryption K (0) -K (n-1) (with contents check value generation key Kicv_cont) Set up one or more contents keys (S1501), and the session key generated at the time of mutual recognition with media 2 is applied. Encryption of data for encryption K (0) - K (n-1) by DES-CBC Mohd is performed using initial value IV_keys stored in the memory section 321, and data K' (0) - K' (n-1) is generated (S1502). This encryption processing is performed in the same processing configuration as drawing 32 explained previously. Next, a memory interface transmits the CBC encryption initialization command to media 2 controller. Media 2 set to a register initial value IV_keys stored in the interior of media 2 (S1506). Then, a memory interface carries out sequential transmission (S1505) of each key.
[0242] Media 2 controller performs decode processing by CBC Mohd by the session key which received data K' (i) (S1507) and was generated to data K' (i) which received at the time of mutual recognition with a device (S1508), and acquires the decoded key data (KONTENTSUKI corresponding to a sector of ex. plurality) (S1509). Next, media 2 controller performs encryption processing by CBC Mohd who used the preservation key Ksto of media 2 for the decode key data stream, generates data stream K" (i), and transmits a result to a device (S1510). Processing of steps S1507-S1510 is performed based on processing by DES-CBC Mohd of drawing 34 who explained previously.
[0243] The memory interface of a device transmits the CBC quit command to media 2 controller after checking having received K" (i) one by one and having received all data (S1511-S1514). Media 2 controller clears a register by reception of the CBC quit

command (S1515).
[0244] K which received the memory interface of a device from media 2 -- let ″(0) - K″ (n-1) be encryption key data for header storing. By the above-mentioned processing, a device can acquire the enciphered contents key Kc which is stored in a header, and contents check value generation key Kicv_cont.
[0245] Explanation of return and encryption write-in processing of a file is continued to drawing 41 . In step S1306, after generation of an above-mentioned header storing key and encryption are completed, a memory interface generates the alteration check value ICV based on the generated header data (S1307). ICV_sh which is the check value of a security header is generated based on the ICV generation configuration previously explained using drawing 14 using the initial value IVsh and security header alteration check value generation key Kicv_sh which were stored in the memory section 321 (refer to drawing 4). Next, processing is ended at step S1308, writing in the generated header, saving inside as a header, and using a busy flag as 0 (standby) for a header generation success flag as 1 (success) at step S1309.
[0246] On the other hand, a control-section side transmits a status read-out command to a memory interface at step S1312, a busy flag is 0 (standby) (S1313), and it reads a header from a buffer a condition [the header generation success flag having been set to 1 (success) (S1314)], and progresses to the next processing (drawing 42) after saving to media (S1315) as a usual file.
[0247] In step S1321 of drawing 42 , a control section divides the contents file for writing into a sector. The divided data are set to D (1) - D (k). A control section sets up write-in sector [of each data D (i)] S (i) next, and carries out sequential transmission (S1321-S1324) of the data D (i) to the encryption write command of sector S (i) at a memory interface. If a sector S(i) encryption write command is received (S1325), a memory interface will set a busy flag as 1 (busy) (S1326), and will progress to degree step a condition [what a header generation success flag is 1 (success) (S1327)].
[0248] Next, receiving sector S (i) judges whether they are an internal memory and external memory (S1328), and when it is external memory, a memory interface Judge (S1329), and when a set flag is 1, whether the set flag of media 1 or media 2 is 1 (it is shown that media are set effectively) Furthermore with reference to a block permission table (BPT), it judges whether BPT wrote in and sector S (i) which is an object is set up as a block for write-in authorization (S1330). When BPT has setting out of a write-in authorization block, the error correcting code set up corresponding to a sector is generated (S1331).

[0249] Next, based on header information (ICV flag), it judges whether the write-in sector is an ICV setting-out sector (S1332), and when it is an ICV object, ICV to sector data is generated based on contents ICV generation key Kicv_cont (S1333).
[0250] Next, a memory interface performs the data encryption based on header information (S1334). The detail of data-division encryption processing of step S1334 is explained using drawing 45 . This encryption processing is performed in the cipher-processing section 320 (refer to drawing 4) of the memory interface of a device.
[0251] First, the data storage sector location for encryption is set to s (0<=s<=31 (in the case with 32 sectors)) (S1601). Next, it is confirmed whether the sector is an object for encryption (S1602). This check is judged based on the encryption flag (Encryption Flag) of a security header (refer to drawing 7). When it is not an object for encryption, encryption processing is not performed but processing is ended. When it is an object for encryption, an encryption format type is checked (S1603). This checks encryption format type (Encryption Format Type) setting out in a security header, and it judges whether the whole contents explained by drawing 8 are made into one encryption mode, or encryption processing using a key which is different into each sector is performed.
[0252] When the encryption format type (Encryption Format Type) set point is 0, it is the case where the whole contents are being made into one encryption mode. In this case, encryption algorithm (Encryption Algorithm) is judged in step S1604. Encryption algorithm has set up Single DES and Triple DES (refer to drawing 28), and when judged with it being Single DES, it performs encryption processing of encryption contents with the application of one contents key Kc (0) (S1606). When judged with it being Triple DES, encryption processing of encryption contents is performed with the application of two contents keys Kc (0) and Kc (1) (S1607).
[0253] It is the case where encryption processing using a key which is different into each sector is performed at step S1603 on the other hand when the code format type (Encryption FormatType) set point is 1. In this case, encryption algorithm (Encryption Algorithm) is judged in step S1605. Encryption algorithm has set up Single DES and Triple DES (refer to drawing 28), and when judged with it being Single DES, it performs encryption processing of encryption contents into each sector with the application of the contents key Kc (s) set up corresponding to each sector (s) (S1608). When judged with it being Triple DES, with the application of two contents keys Kc and Kc (s+1mod32), encryption processing for every sector is performed (s) (S1609).
[0254] The processing mode from which decode processing of sector data differs is

shown in drawing 46 . In drawing 46 , steps S1701-S1708 are the same as each steps S1601-S1608 of drawing 45 . Steps S1709-S1711 differ from drawing 45 .

[0255] In step S1705, when judged with encryption algorithm being Triple DES, sector No. (s) is judged in step S1709, when the number of s is odd, renewal of s=s -1 is performed (S1710), and decode processing (S1711) according the key applied to each sector to Triple DES is performed as (s), Kc, and Kc (s+1).

[0256] Explanation of return and the encryption write-in processing flow of a file is continued to drawing 42 . After the encryption processing step (S1334) of data division is completed by above-mentioned processing, the error correcting code to data division is generated (S1335), the alteration check value ICV corresponding to Data D (i) and sector data which were enciphered, and the redundancy section with an error correcting code are written in media (S1336), a write-in success flag is set to 1 (success) (S1337), and a busy flag is set as 0 (standby) (S1339).

[0257] In addition, when the data for writing are write-in processing into the internal memory by which management by BPT is not made, steps S1329 and S1330 are skipped. When the judgment of steps S1329 and S1330 is No (i.e., when the set flag of media is not 1), or when write-in authorization of sector S (i) is not set as BPT, it progresses to step S1338, and writes in as a write error, and a success flag is set to 0.

[0258] moreover, a control section carries out reading appearance of the status of a memory interface in steps S1341-S1345, and in the condition of 0, a busy flag carries out sequential increment of the address a condition [a write-in success flag being 1], and transmits write-in data to a memory interface one by one. After all processings are completed, an update process of a file quota table is performed (S1346), the updated file quota table is transmitted to a memory interface with an updating command (S1347), and a memory interface performs write-in processing of a file quota table according to a command (S1340).

[0259] Storing processing to a data encryption and media is performed by processing explained by the above drawing 41 - drawing 46 .

[0260] An update process of [renewal of a RIBOKESHON list], next the RIBOKESHON list as annulment information on inaccurate media or contents is explained. As mentioned above, the RIBOKESHON list in this invention consists of identifiers (ID) of two or more classes (ex. media, contents). It becomes possible to eliminate the contents of two or more classes by one RIBOKESHON list, and media by preparing ID of two or more classes in the RIBOKESHON list (Revocation List) which is the annulment information on contents or media, and performing each collating as different actuation. The activity of inaccurate media and read-out of inaccurate

contents can be forbidden by performing collating with the identifier (ID) of utilization media or utilization contents, and the listing ID of a RIBOKESHON list in the memory interface section at the time of insertion of media, and read-out of contents.
[0261] A RIBOKESHON list is updated, when a RIBOKESHON list version (Revocation List Version) is set to a RIBOKESHON list and the annulment information on new inaccurate media or contents is added to it, as explained previously.
[0262] The update process flow of a RIBOKESHON list is shown in drawing 47 . In drawing 47 , left-hand side is the control section of a device, and right-hand side is the memory interface of a device.
[0263] first, the RIBOKESHON list for updating in a control section -- from the communications department 201 (refer to drawing 2) -- receiving (S1801) -- the RIBOKESHON list for updating received with the RIBOKESHON list check command for updating is transmitted to a memory interface (S1802).
[0264] If the RIBOKESHON list for updating is received from a control section with the RIBOKESHON list check command for updating (S1803), a memory interface will set a busy flag as 1 (busy) (S1804), and will generate alteration check value (ICV) generation key Kicv_rl of a RIBOKESHON list (S1805).
[0265] Alteration check value (ICV) generation key Kicv_rl for the alteration check of a RIBOKESHON list is generated based on the RIBOKESHON list version (Version) contained in master key:MKicv_rl which generates the ICV key of the RIBOKESHON list (Revocation List) beforehand stored in the device, initial value:IVicv_rl when generating the ICV key of a RIBOKESHON list (Revocation List), and the attribute information on a RIBOKESHON list. Specifically based on alteration check value (ICV) generation key Kicv_rl=DES (E, MKicv_rl, Version^IVicv_rl), an alteration check value (ICV) generation key is generated. The semantics of said formula means that encryption processing by DES Mohd by master key:MKicv_rl is performed to the exclusive OR of a version (Version) and initial value (IVicv_rl).
[0266] Next, using generated alteration check value (ICV) generation key Kicv_rl, it generates (S1806) and a memory interface performs the right ICV value beforehand stored in the RIBOKESHON list, and collating ICV'=ICV? for ICV' of a RIBOKESHON list (S1807). In addition, generation processing of ICV' is performed by the processing which applied generated alteration check value (ICV) generation key Kicv_rl using initial value IVrl based on DES Mohd who explained by above-mentioned drawing 14 .
[0267] When it is ICV'=ICV (it is Yes at S1807) It is judged with it being a just thing without an alteration of the RIBOKESHON list for updating. The version (i) of the RIBOKESHON list by which the current set is carried out by progressing to step

S1808 is compared with the version (j) of the RIBOKESHON list for updating (S1809). When the version of the RIBOKESHON list for updating is new, the effective flag of the RIBOKESHON list for updating is set as 1 (S1810), a busy flag is set to 0 (S1811), and processing is ended.

[0268] On the other hand, a control-section side transmits a status read-out command to a memory interface (S1812), checks what (S1813) the busy flag was set to 0 for, and when the RIBOKESHON list effective flag for updating is 1 (S1814), it saves the RIBOKESHON list for updating as a usual file at an internal memory (S1815). In the case of the check at the time of processing of contents, and wearing of media, reading appearance of the RIBOKESHON list stored in the internal memory is carried out.

[0269] As mentioned above, it has explained in detail about this invention, referring to a specific example. However, it is obvious that this contractor can accomplish correction and substitution of this example in the range which does not deviate from the summary of this invention. That is, with the gestalt of instantiation, this invention has been indicated and it should not be interpreted restrictively. In order to judge the summary of this invention, the column of the claim indicated at the beginning should be taken into consideration.

[0270]

[Effect of the Invention] As mentioned above, since it considered as the configuration which sets version information as a RIBOKESHON list according to the data regenerative apparatus of this invention, the data recorder and the data playback approach, the data-logging approach, and the renewal approach of a list as explained For example, in the case of contents read-out, the version of the RIBOKESHON list currently held to the current device is compared with the version of the effective RIBOKESHON list in the header of contents. When the version of the RIBOKESHON list which is carrying out current maintenance is older, the processing which stops read-out of contents is attained. Consequently, if a RIBOKESHON list is not updated, read-out of the contents can be performed and can eliminate the unauthorized use of the contents which used the old RIBOKESHON list.

[0271] Furthermore, according to the data regenerative apparatus of this invention, a data recorder and the data playback approach, the data-logging approach, and the renewal approach of a list The RIBOKESHON list for updating received, for example from a channel also in an update process of a RIBOKESHON list, Since it considered as the configuration to which renewal of a RIBOKESHON list is permitted only when it judged that the version information of a current RIBOKESHON list is compared and the list for updating is a newer RIBOKESHON list It becomes possible to prevent processing of being unjustly transposed to an old list.

DESCRIPTION OF DRAWINGS

[Brief Description of the Drawings]
[Drawing 1] It is drawing explaining the activity concept of the data processor of this invention.
[Drawing 2] It is drawing showing the device of the data processor of this invention, and the configuration of media.
[Drawing 3] It is drawing showing the memory storing data configuration of the data processor of this invention.
[Drawing 4] It is drawing showing the detail configuration of the memory interface of a device in the data processor of this invention.
[Drawing 5] It is drawing showing the data configuration of the status register of the memory interface in the data processor of this invention.
[Drawing 6] It is drawing showing the detail configuration of the data stored in the media in the data processor of this invention.
[Drawing 7] It is drawing explaining the configuration of the security header set up corresponding to the contents stored in media in the data processor of this invention.
[Drawing 8] It is drawing explaining two modes of the data encryption in the data processor of this invention.
[Drawing 9] It is drawing showing the configuration of the RIBOKESHON list in the

data processor of this invention.
[Drawing 10] It is drawing explaining the block permission table (BPT) in the data processor of this invention.
[Drawing 11] It is drawing showing the BPT storing processing flow at the time of the media 1 manufacture in the data processor of this invention.
[Drawing 12] It is drawing showing the BPT storing processing flow at the time of the media 2 manufacture in the data processor of this invention.
[Drawing 13] It is drawing explaining the example of the block permission table (BPT) in the data processor of this invention.
[Drawing 14] It is drawing explaining the alteration check value generation processing configuration in the data processor of this invention.
[Drawing 15] It is drawing explaining the alteration check value verification processing flow in the data processor of this invention.
[Drawing 16] It is drawing showing a flow at the time of device starting in the data processor of this invention.
[Drawing 17] It is drawing explaining the example of a configuration of the file quota table in the data processor of this invention.
[Drawing 18] It is drawing showing a flow (the 1) at the time of the media 1 recognition in the data processor of this invention.
[Drawing 19] It is drawing showing a flow (the 2) at the time of the media 1 recognition in the data processor of this invention.
[Drawing 20] It is drawing showing a flow (the 1) at the time of the media 2 recognition in the data processor of this invention.
[Drawing 21] It is drawing showing a flow (the 2) at the time of the media 2 recognition in the data processor of this invention.
[Drawing 22] It is drawing showing the mutual recognition processing sequence performed between device media in the data processor of this invention.
[Drawing 23] It is drawing showing the mutual recognition and the key share processing flow in the data processor of this invention (the 1).
[Drawing 24] It is drawing showing the mutual recognition and the key share processing flow in the data processor of this invention (the 2).
[Drawing 25] It is drawing showing the read-out processing flow of the file in the data processor of this invention.
[Drawing 26] It is drawing showing the write-in processing flow of the file in the data processor of this invention.
[Drawing 27] It is drawing explaining the data encryption processing mode stored in

the memory in the data processor of this invention.
[Drawing 28] It is drawing explaining Triple DES applicable as a data encryption processing mode stored in the memory in the data processor of this invention.
[Drawing 29] It is drawing explaining the data encryption processing mode stored in the memory in the data processor of this invention.
[Drawing 30] It is drawing explaining the data encryption processing mode stored in the memory in the data processor of this invention.
[Drawing 31] It is drawing explaining the storing processing mode of the alteration check value corresponding to a sector in the data processor of this invention.
[Drawing 32] It is drawing explaining the example of encryption processing of a key besides a contents key corresponding to a sector in the data processor of this invention.
[Drawing 33] It is drawing explaining the example of decode processing of a key besides a contents key corresponding to a sector in the data processor of this invention.
[Drawing 34] It is drawing explaining the example of processing between the device media of a key besides the contents key corresponding to a sector in the data processor of this invention.
[Drawing 35] It is drawing showing the decode read-out processing flow (the 1) of the file in the data processor of this invention.
[Drawing 36] It is drawing showing the decode read-out processing flow (the 2) of the file in the data processor of this invention.
[Drawing 37] It is drawing showing the decode processing flow besides a contents key in the data processor of this invention.
[Drawing 38] It is drawing showing a decode processing flow with the preservation key of media besides a contents key in the data processor of this invention.
[Drawing 39] It is drawing showing the decode processing flow (the 1) of the sector data in the data processor of this invention.
[Drawing 40] It is drawing showing the decode processing flow (the 2) of the sector data in the data processor of this invention.
[Drawing 41] It is drawing showing the encryption write-in processing flow (the 1) of the file in the data processor of this invention.
[Drawing 42] It is drawing showing the encryption write-in processing flow (the 2) of the file in the data processor of this invention.
[Drawing 43] It is drawing showing the encryption processing flow besides a contents key in the data processor of this invention.

[Drawing 44] It is drawing showing an encryption processing flow with the preservation key of media besides a contents key in the data processor of this invention.
[Drawing 45] It is drawing showing the sector data encryption processing flow (the 1) in the data processor of this invention.
[Drawing 46] It is drawing showing the sector data encryption processing flow (the 2) in the data processor of this invention.
[Drawing 47] It is drawing showing the update process flow of the RIBOKESHON list in the data processor of this invention.
[Description of Notations]

101	System Management Person
102	Device
103	Media
200	Device
201	Communications Department
202	Input Section
203	Display
204	Device Controller
205	Control Section
207	Memory Section
300	Memory Interface (I/F) Section
210	Media 1
211	Control Section
212	Memory Section
230	Media 2
231	Controller
232	Memory Section
233	Control Section
234	Memory Interface (I/F) Section
235	Internal Memory
236	Cipher-Processing Section
301	Status Register
302	Command Register
303	Address Register
304	Count Register
305	Control Register
306	Transmit/receive Control Section

307 Transmitting Buffer Memory
308 Receiving Buffer Memory
309 Transmit Register
310 Receive Register
320 Cipher-Processing Section
321 Memory Section
323 ECC Circuit
324 External Memory Input/output Interface
325 Internal-Memory Input/output Interface